[illegible]

05952-6227-R0-00

TRW NOTE NO. 68-FMT-700

PROJECT APOLLO
TASK MSC/TRW A-116.5

A COMPUTER PROGRAM TO PREDICT THE IMPACT DISPERSIONS OF DEBRIS RESULTING FROM THE FORCED ENTRY OF APOLLO VEHICLES

4 OCTOBER 1968

Prepared by
Engineering Mechanics Laboratory
TRW Systems Group

Prepared for
MISSION PLANNING AND ANALYSIS DIVISION
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
MANNED SPACECRAFT CENTER
HOUSTON, TEXAS
NAS 9-4810

(NASA-CR-133317) A COMPUTER PROGRAM TO PREDICT IMPACT DISPERSIONS OF DEBRIS RESULTING FROM THE FORCED ENTRY OF APOLLO VEHICLES (TRW Systems Group) 104 p

N73-73151

Unclas
00/99 17955

05952-6227-R0-00

TRW NOTE NO. 68-FMT-700

PROJECT APOLLO

TASK MSC/TRW A-116.5

A COMPUTER PROGRAM TO PREDICT IMPACT DISPERSIONS OF DEBRIS RESULTING FROM THE FORCED ENTRY OF APOLLO VEHICLES

4 OCTOBER 1968

Prepared for
MISSION PLANNING AND ANALYSIS DIVISION
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
MANNED SPACECRAFT CENTER
HOUSTON, TEXAS
NAS 9-4810

Prepared by [illegible]
M. H. Marx

Approved by L Wilcox
L. F. Wilcox, Manager
Task MSC/TRW A-116

Approved by [illegible]
J. H. Walker, Manager
Dynamics Department

Approved by C. R. Coates
C. R. Coates
Assistant Project Manager
Flight Dynamics
Mission Trajectory Control
Program

TRW
SYSTEMS GROUP

ACKNOWLEDGMENT

The author wishes to thank Mr. Michael T. Palmiter for his formulation of the reentry equations as presented in Appendix D of this report,and Mr. Clayton E. Nenno for incorporating them into the Monte Carlo program. This "Patched Analytic" technique (applied to non-lifting reentry) was first conceived in an internal research project by the Vehicle Analysis Department (E. J. Der, A. M. Mukhey) of TRW Systems Group during 1967-1968.

The author also wishes to thank Mr. Garth W. Lippmann for the aerodynamic analysis of various debris types as presented in Appendix C.

ABSTRACT

A computer program is developed which uses a Monte Carlo simulation to predict the impact distributions for forced entry debris. Required input is a complete state-vector at the vehicle breakup and a list of the surviving debris. All piece aerodynamic parameters are randomly selected and a very rapid analytic reentry subroutine combined with an empirical formulation for non-lifting trajectories yields a highly efficient program which can be used for any Apollo mission.

CONTENTS

ILLUSTRATIONS

1. INTRODUCTION AND SUMMARY

The purpose of Task MSC/TRW A-116 is to evaluate the hazards to people on the earth as a result of the reentry of specific Apollo vehicles and/or components. This technical report is submitted to the Manned Spacecraft Center by TRW Systems Group in accordance with Subtask MSC/TRW A-116.5 of the Apollo Mission Trajectory Control Program.

A previous study (References 1 and 2) examined new techniques which lead to a forced entry impact dispersion analysis for debris resulting from the breakup of the Apollo 6 Service Module. The results of that study indicated that prior worst case analyses were extremely conservative and provided answers which were very unrealistic.

A computer program was developed during that study which was based on these new techniques and offered a number of significant improvements over the previous methods used for this type of analysis. However, the program developed at that time was mission oriented in that it contained certain empirical formulations which had been derived specifically for Apollo lunar returns similar to those entry conditions experienced during the Apollo 6 mission.

Since the analytical techniques which had been developed were equally valid for other missions it became desirable to transform this mission oriented program to one which would have a more general validity. The effort was primarily one of replacing a parametric-empirical approach toward the reentry computations with a more analytic technique which would provide the necessary flexibility to handle a variety of missions.

This report documents a new computer program developed under Subtask A-116.5 which uses the basic techniques presented in References 1 and 2 but provides the flexibility for use with any Apollo mission. This new program will give the Apollo forced entry debris effort a means for determining debris impact dispersions with improved accuracy, fast response time, and high flexibility.

This report is a republication, in part, of Reference 1 and is intended to replace that document. The basic background and unchanged portions of the previous study have been reproduced in this report for the sake of clarity and completeness of this document.

2. METHOD OF ANALYSIS

Prior Gemini and Apollo orbital debris hazard studies have recognized that the possibility of lift associated with the various pieces of debris produced the greatest impact uncertainty of all the significant parameters. That previous techniques for the evaluation of the entry debris impact dispersions have led to over estimation of the impact probability for points remote from the intended impact area is primarily a result of the manner in which this aerodynamic lift has been handled.

The effect of lift was maximized by assuming that:

- All fragments would assume worst trim attitudes, resulting in a maximum lift/drag ratio (L/D),
- All fragments would be of worst shape, having the the largest L/D to be expected for fragments of various shapes,
- All fragments would fly at constant bank angle, maximizing deviation from the predicted zero lift impact point.

These worst case conditions led to maximum downrange deviations of nearly 18,000 n mi and crossrange deviations of $\pm$ 600 n mi from the nominal (zero lift) impact point (Reference 3). The manner of assigning a distribution for the impact locations of the surviving pieces was then to assume a split bivariate normal distribution as sketched in Figure 1.

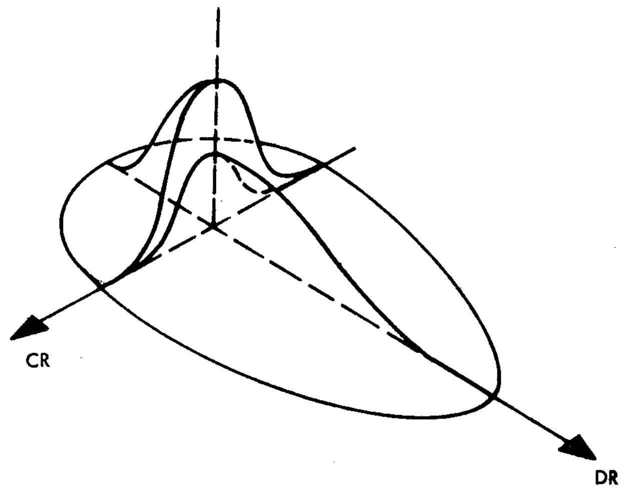


Figure 1. Split Bivariate Normal Distribution

The maximum deviations in the uprange, downrange, and cross-range directions were then taken as 3 sigma (3σ) values. All the shape characteristics of this distribution are normal, but the standard deviations uprange and downrange are not equal.

While this manner of treating the impact distributions may have maximized the effect of lift on the various pieces of debris, the actual hazard computation is not necessarily a conservative estimate of the impact probability. Since the integrated volume under a probability density function must remain equal to 1, the fact that the density function has been spread out to such a large extent results in a significantly decreased density in the vicinity of the nominal impact point. Therefore, the probability of impact within 6000 miles of the nominal impact point (1σ) will be greatly underestimated, while the probabilities of impact at remote points will be significantly overestimated.

The purpose of Subtask A-116.5 was to develop a computer program which would provide a more meaningful statistical description of the impact distributions for forced entry debris. The problem of lift had to be treated in a more rational fashion and techniques were required to provide a best estimate of the actual aerodynamic characteristics of a particular piece of debris. In an earlier study (References 1 and 2) such a program was developed for the analysis of the debris resulting from a breakup of the Service Module during the Apollo 6 mission. However, this program was oriented toward a specific mission analysis. Impact location for a piece of debris was determined empirically from parametric data relating impact location to the aerodynamic properties of the debris for the particular state vector at the time of the Service Module breakup.

While this approach was the most expedient for a particular mission analysis, to utilize this program for any other mission with different entry conditions would require a reformulation of the parametric relationships and a certain amount of modification to the program.

The objective of the present study was to apply these same techniques to develop a more flexible program which could handle a broad variety of missions and be cost effective compared to the parametric-empirical approach over an intermediate or long term basis. The only necessary change for the program which existed was to replace the empirical determination of impact locations by an analytic reentry computation scheme which would not be reliant on any previously formulated relationships for the specific entry conditions.

This report documents the new Apollo Forced Entry Debris Dispersion Program. While certain other changes and refinements have been made,the new program is basically the same as that described in Reference 1 with the major change being the analytic computation of impact location from a set of given initial conditions. Since the nature of a Monte Carlo analysis (see Section 3. 2) can require this computation to be performed many thousands of times, a great effort had to be made to produce a very rapid reentry solution. Otherwise, the cost of computing time required for a mission analysis would become prohibitive.

The advantages of such an improved program are:

- flexibility - The program can analyze any mission, given the state vector at breakup and the nature of the debris.
- fast response time - There is no need for a parametric study and program modification prior to its use for a specific mission.
- improved accuracy - There is no reliance on "best fit" parametric relationships.

3. DESCRIPTION OF THE APOLLO FORCED ENTRY DEBRIS DISPERSION COMPUTER PROGRAM

3.1 POSSIBLE APPROACHES TO THE PROBLEM

The ideal manner of determining an impact dispersion for a particular piece of debris would be to integrate the six-dimensional motion of the piece to an impact point for each possible combination of its time varying orientation and angular rates, aerodynamic properties, and trajectory initial conditions. The impact probability would then simply be a function of the distribution of all such impact points. A prerequisite to this approach is a detailed knowledge of the aerodynamic properties of the particular piece. However, it is virtually impossible to realistically determine for an arbitrary and ill-defined piece of debris (in the the actual shape, c.g. location, c.p. location, mass properties, etc., are unknown) the aerodynamic coefficients for lift, drag, roll, and damping. Without these data a six-dimensional analysis would prove rather meaningless.

With the necessary simplification to a three-dimensional model, the most exacting approach would be the consideration of all possible combinations of time varying motion for each piece of debris. However, the vast number of possible states for each piece of debris and the difficulty of dealing with time varying parameters makes this approach also unfavorable.

The approach used in this study was to treat each reentering piece as a mass whose three-dimensional motion is determined by an initial state vector, a set of constant parameters (lift/drag ratio (L/D), ballistic coefficient (W/C_DA), an initial bank angle (θ_i), and an initial bank angle rate ($\dot{\theta}_i$)), and a bank angle rate ($\dot{\theta}$) which varies along the reentry trajectory. Although it is rather unlikely that a piece would maintain such a state throughout its entire reentry, this assumption was necessary to keep the magnitude of the task within reason, and for most cases it should result in a conservative estimate of the actual motion in that a time varying parameter probably tends to average out to some nominal value.

3.2 MONTE CARLO SIMULATION

A Monte Carlo simulation is a technique for predicting the behavior of a system by randomly sampling the system's components, computing the system's behavior for this combination of components, repeating this process a large number of times, and then (based on these data) predicting the probable behavior of the system. This procedure is shown graphically in Figure 2. In essence, the use of a Monte Carlo simulation is similar to an actual flight test program in which a large number of flights leads to a prediction of the system's average behavior and distribution of possible behavior. The availability of high speed computers, which can economically and rapidly synthesize the performance of complex systems has led to the popularization of Monte Carlo procedures.

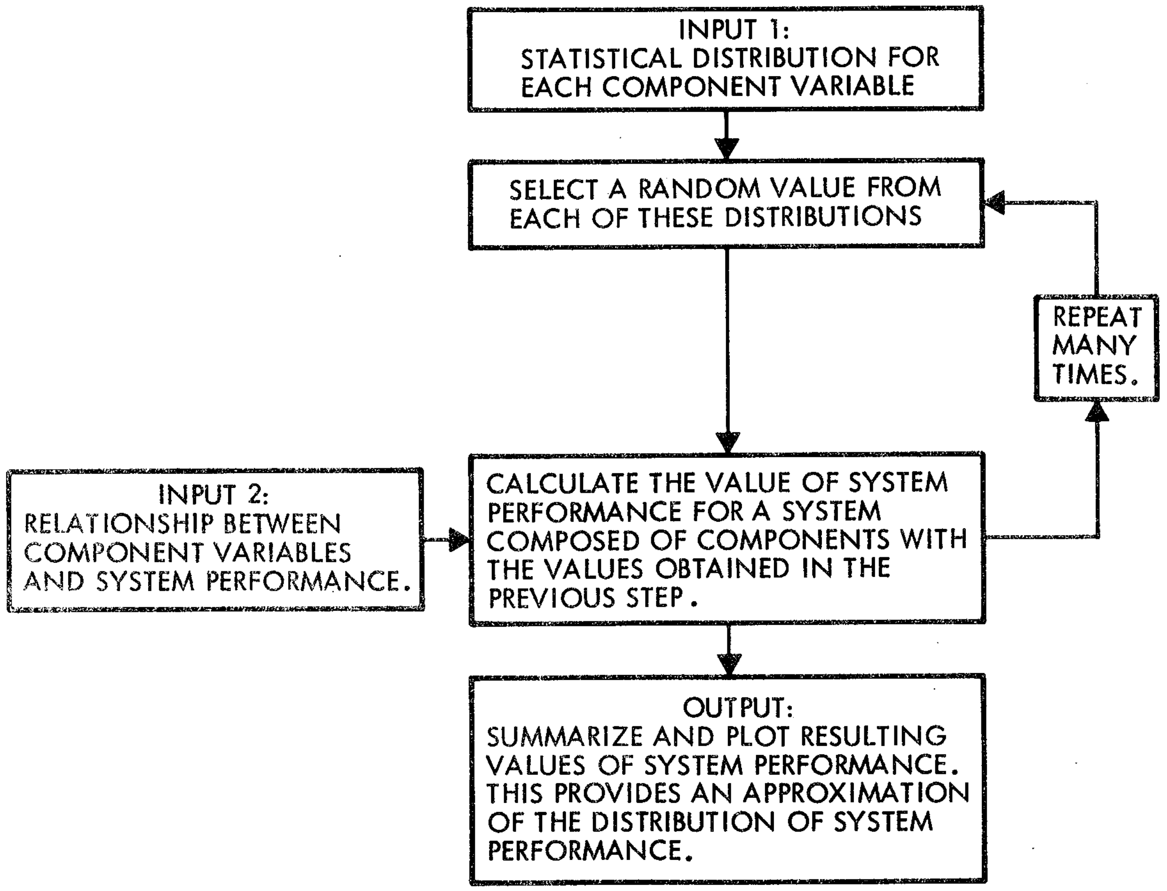

INPUT 1:
STATISTICAL DISTRIBUTION FOR EACH COMPONENT VARIABLE
SELECT A RANDOM VALUE FROM EACH OF THESE DISTRIBUTIONS
REPEAT MANY TIMES.
INPUT 2:
RELATIONSHIP BETWEEN COMPONENT VARIABLES AND SYSTEM PERFORMANCE.
CALCULATE THE VALUE OF SYSTEM PERFORMANCE FOR A SYSTEM COMPOSED OF COMPONENTS WITH THE VALUES OBTAINED IN THE PREVIOUS STEP.
OUTPUT:
SUMMARIZE AND PLOT RESULTING VALUES OF SYSTEM PERFORMANCE. THIS PROVIDES AN APPROXIMATION OF THE DISTRIBUTION OF SYSTEM PERFORMANCE.

Figure 2. Flow Chart of Monte Carlo Simulation Method

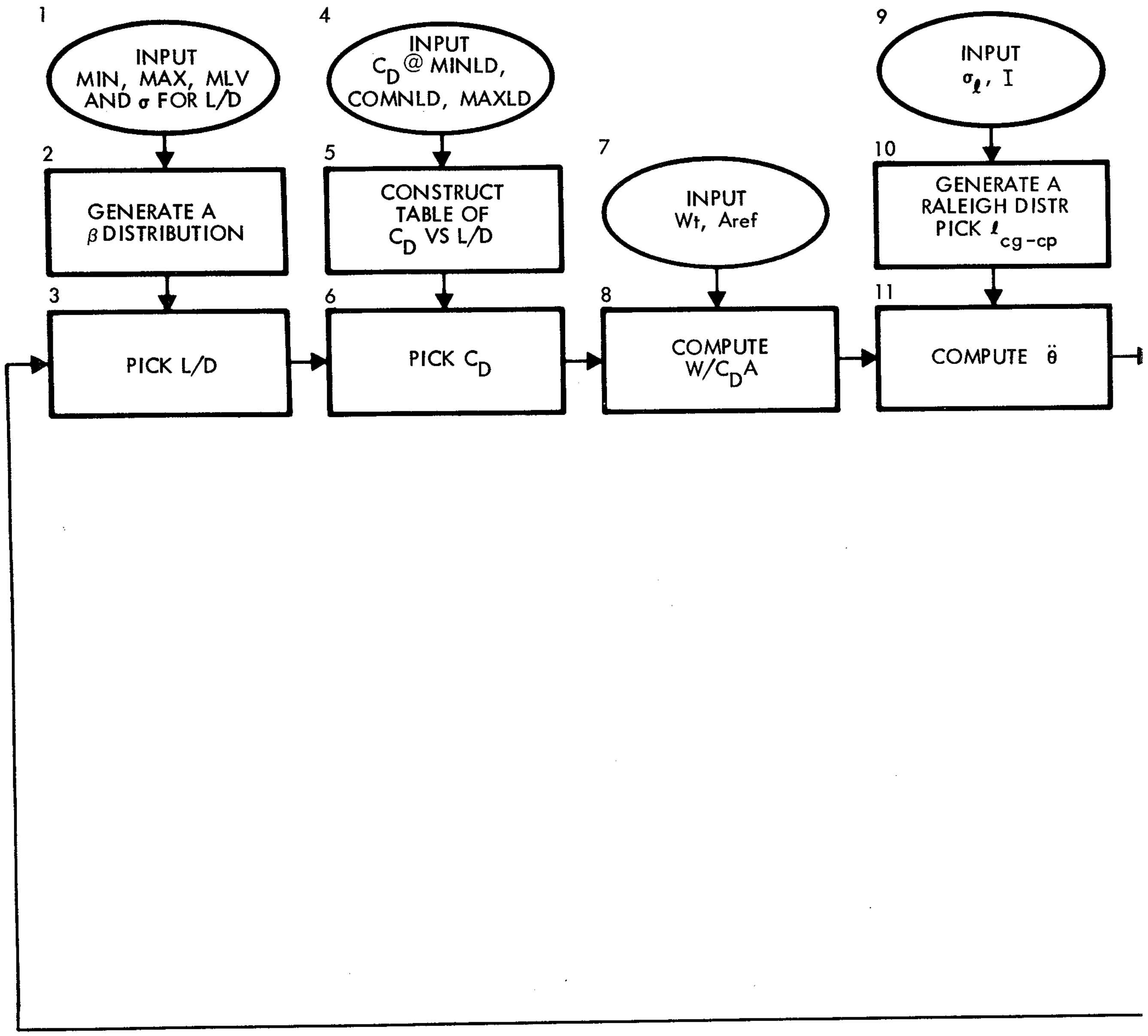
3.3 OUTLINE OF THE PROGRAM

As stated earlier, the end objective of this study was to formulate a computer program which could predict the impact distribution for the various debris types and entry conditions resulting from a variety of Apollo missions. For any particular mission analysis, the Monte Carlo technique described in the previous section is used to perform this task as outlined by the program flow diagram in Figure 3. Since each distinct debris piece type has a unique impact distribution and an associated lethal area, the program is fully executed for each of these types of debris and the entire mission can be analyzed as a series of cases.

The following pages outline the operation of the program. The numbered sections explain the individual elements in the program flow diagram which bear the corresponding number. This sequence of elements does not always reflect the order of occurrence within the program but is presented in this fashion for clarity.

1. The first necessary input to the program is the minimum, maximum, most likely value, and standard deviation for the L/D of the particular piece. The standard deviation, σ_β, may be omitted as explained in element 2. The methods used to determine these and all other aerodynamic input data for the program are described in detail in Appendix C.

2. For L/D the theoretical maximum might be as high as 2.5 for some of the piece types. However, the probability of such a state occurring during the entire reentry would be extremely unlikely. Therefore, it was decided to statistically describe the probability of occurrence of L/D by a highly skewed distribution with finite bounds. The Beta distribution is best suited to this purpose being defined by a minimum, a maximum, a most likely value, and a standard deviation.


1
INPUT MIN, MAX, MLV AND σ FOR L/D
2
GENERATE A β DISTRIBUTION
3
PICK L/D
4
INPUT C_D @ MINLD, COMNLD, MAXLD
5
CONSTRUCT TABLE OF C_D VS L/D
6
PICK C_D
7
INPUT Wt, Aref
8
COMPUTE W/C_DA
9
INPUT σ_ℓ, I
10
GENERATE A RALEIGH DISTR PICK ℓ_{cg-cp}
11
COMPUTE $\ddot{\theta}$

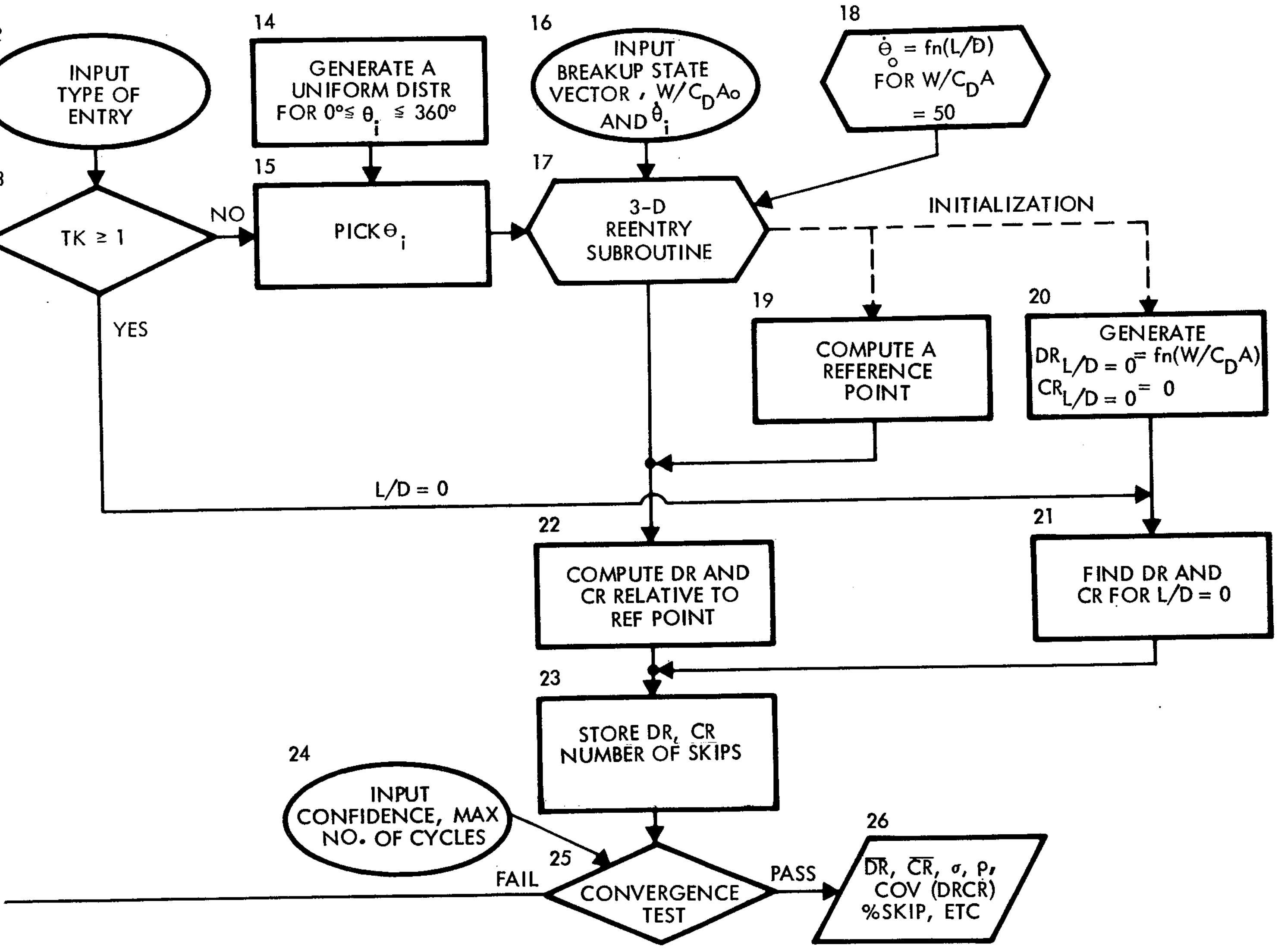
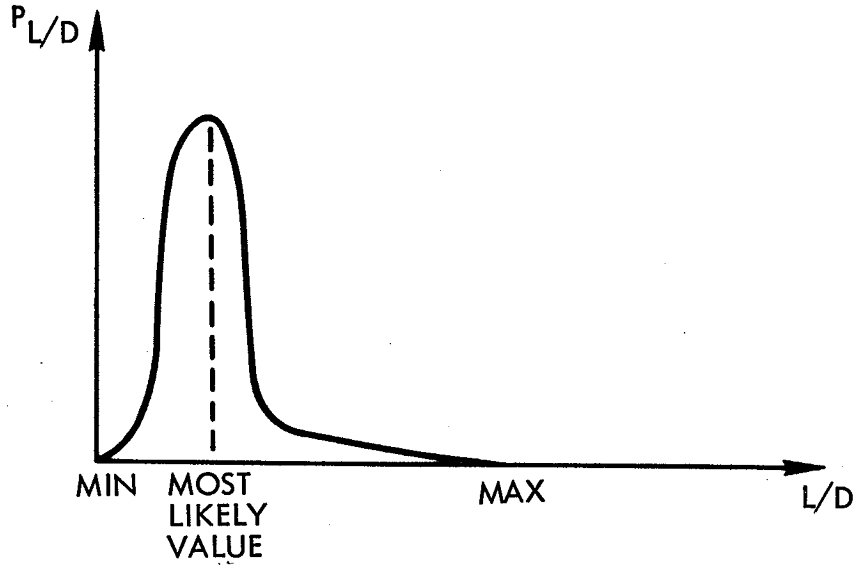

INPUT TYPE OF ENTRY
14
GENERATE A UNIFORM DISTR FOR 0° ≤ θi ≤ 360°
16
INPUT BREAKUP STATE VECTOR, W/CDAo AND θ̇i
18
θ̇o = fn(L/D) FOR W/CDA = 50
15
17
NO
TK ≥ 1
PICK θi
3-D REENTRY SUBROUTINE
INITIALIZATION
YES
19
COMPUTE A REFERENCE POINT
20
GENERATE DR L/D = 0 = fn(W/CDA) CR L/D = 0 = 0
L/D = 0
22
21
COMPUTE DR AND CR RELATIVE TO REF POINT
FIND DR AND CR FOR L/D = 0
23
STORE DR, CR NUMBER OF SKIPS
24
INPUT CONFIDENCE, MAX NO. OF CYCLES
25
FAIL
CONVERGENCE TEST
PASS
26
DR̄, CR̄, σ, ρ, COV (DRCR) %SKIP, ETC

Figure 3. Flow Diagram of the Monte Carlo Program



The Beta distribution is defined as follows:

$$p(x) = Ax^a (1 - x)^b \text{ for } 0 < x < 1$$

$$= 0 \text{ elsewhere}$$

where

$$A = \frac{\Gamma(a + b + 2)}{\Gamma(a + 1) \cdot \Gamma(b + 1)}$$

and where a and b are related to the most likely value (ζ) and standard deviation given in element 1 by:

$$\zeta = \frac{a}{a + b}$$

$$\sigma_\beta = \frac{(a + 1)\ (b + 1)}{(a + b + 2)^2 (a + b + 3)}$$

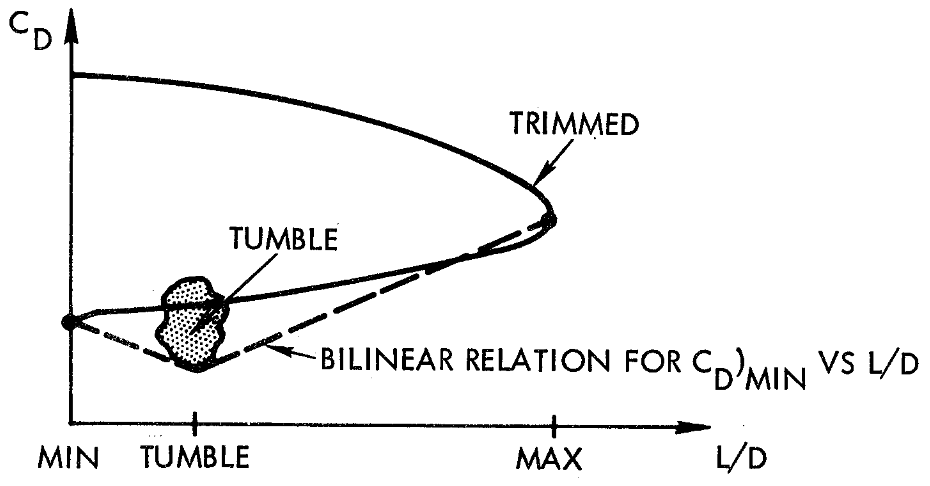
If the standard deviation is not given in element 1, then a is set equal to 1 in the program.

The actual distribution must be normalized to lie in the range $0 \leq L/D \leq 1$ and then the selected value of L/D is corrected to reflect its true value.

3. From the Beta distribution the program will randomly select a value for L/D.

4. The second necessary set of input parameters are the minimum value of C_D at L/D) min, the minimum value of C_D corresponding to a tumbling configuration, and the value of C_D at L/D) max.

5. A consideration of the true aerodynamics of an object which may either trim at some attitude or tumble yields a relation between C_D and L/D of the form



The continuous curve represents a double-valued relationship between trimmed C_D and L/D based on a consideration of a range of angles of attack between 0^o and 90^o. The shaded region represents a range of C_D's corresponding to various tumbling states (see Appendix C).

In order to produce the most conservative analysis, one should choose the minimum value of C_D corresponding to any given value of L/D. This in turn leads to a maximum ballistic coefficient and therefore, the largest dispersion. To this end, the program constructs a bilinear table for C_D vs L/D based on the minimum values input in element 4 and represented by the broken lines in the figure above.

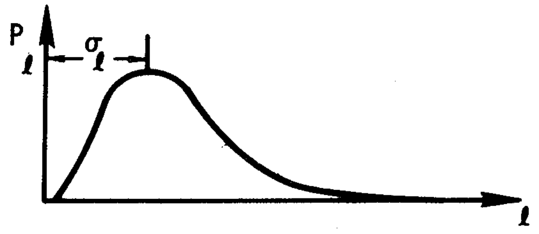
6. A value of C_D is selected corresponding to that value of L/D chosen in element 3.

7. Input the weight (lb) and reference area (ft^2) for the piece (see Appendix C).

8. The program computes a ballistic coefficient, W/C_DA, for the piece based on the selected value of C_D and the inputs from 7.

9. Input the spin moment of inertia for the piece, I($slug\text{-}ft^2$), and the standard deviation for the center of gravity— center of pressure separation, σ_ℓ (ft) (if σ_ℓ is not specified it is set equal to 0.01 in the program).

10. For a given piece of debris, the center of pressure and the center of gravity may be separated by an unknown distance, ℓ. Assuming a bivariate normal distribution for the center of gravity - center of pressure separation in a plane normal to the body spin axis gives a Rayleigh distribution for absolute separation, ℓ.



The net resultant lift vector acts through the center of pressure and creates a torque about the center of gravity which produces a bank angle rate, $\dot{\theta}$. This bank angle rate is a function of the magnitude of the torque acting about the center of gravity and the length of time for which the torque acts. The equation of motion is

$$I\ddot{\theta} = T = L\ell \sin\theta$$

assuming an average torque, $\bar{T} = (2/\pi)\ell L$, then

$$\ddot{\theta} = \frac{2}{\pi}\frac{\ell}{I} L$$

$$= \frac{2}{\pi}\frac{\ell}{I}\left\{\frac{L}{D} \times \frac{1}{2}\rho V^2 \frac{W}{A(W/C_D A)} A\right\}$$

$$= \frac{2}{\pi}\frac{\ell}{I}\left\{\frac{L}{D} \times \frac{1}{2}\rho_o e^{-\frac{h}{23500}} V^2 \frac{W}{A(W/C_D A} A\right\}$$

and with $\rho_o = 2.378 \times 10^{-3}$ slug/ft^3

$$\ddot{\theta}\,({}^{o}/\mathrm{sec}^2) = \left\{0.04336\,\frac{\ell}{I}\,(L/D)\,\frac{W}{(W/C_D A)} V^2 e^{-\frac{h}{23500}}\right\}$$

where

ℓ =	selected c.g. - c.p. separation, ft	
I =	spin moment of inertia, slug - ft^2	
L/D =	selected lift/drag ratio	
W =	weight, lb	
$W/C_D A$ =	computed ballistic coefficient, lb/ft^2	
V =	initial velocity, ft/sec	(see element 16)
h =	initial altitude, ft	

11. The program randomly selects a value for ℓ from the Rayleigh distribution and then computes a bank angle acceleration, $\ddot{\theta}$.

12. The type of entry is input as either typical of an earth orbital mission or a lunar return.

13. If the bank angle acceleration is sufficiently large, the resulting bank angle rate will rapidly build up to a level at which the lift vector rotates so quickly through 360° that the average L/D = 0. For the type of entry typical of a lunar return this "critical bank angle acceleration" is conservatively taken as 5°/sec^2 while for the higher, shallower entry typical of an earth orbital return a value of 0.5°/sec^2 is used since the lift during the early portion of reentry will be much smaller. The program generates a test constant, TK, given by

$$TK = \ddot{\theta}\left[\frac{10^{\text{Entry}}}{50}\right]$$

where

$$TK \geq 1 \text{ for} \begin{cases} \text{ENTRY} = 1 \text{ and } \ddot{\theta} \geq 5^{\circ}/\text{sec}^2 \\ \quad \text{or} \\ \text{ENTRY} = 2 \text{ and } \ddot{\theta} \geq 0.5^{\circ}/\text{sec}^2 \end{cases}$$

If TK ≥ 1 the effective L/D will equal 0 and the program makes a bypass to element 21. Otherwise, it computes a bank angle rate factor

$$\kappa = 57.3\left(\frac{2}{\pi}\right)\frac{\ell}{I}\left(\frac{W}{32.2}\right)$$

and proceeds to element 14.

14. The occurrence of the initial lift vector orientation, θ_i, is considered equally probable for any angle. Therefore, a uniform distribution is generated for $0^{\circ} \leq \theta_i \leq 360^{\circ}$.

15. The program randomly picks a value for θ_i from the distribution in element 14.

16. The next required input to the program is the complete state vector at breakup - altitude (ft), inertial velocity (ft/sec), inertial flight path angle (deg), inertial azimuth (deg), latitude (deg), and longitude (deg, positive east of Greenwich), the ballistic coefficient corresponding to the desired reference point, $(W/C_DA)_o$ - typically the ballistic coefficient for the whole vehicle, and an initial bank angle rate, $\dot{\theta}_i$ ($^{\circ}$/sec) (if $\dot{\theta}_i$ is not input it is set = 0 by the program).

17. The major subroutine within the program is one which analytically computes the reentry trajectory. The state vector at breakup (element 16) and the piece aerodynamic parameters (elements 3, 8, 15, 16) are supplied to the subroutine and it then computes a downrange and cross-range miss distance in nautical miles for the impact point of that piece relative to a reference point (see element 19). If the particular combination of input parameters results in the piece skipping out of the atmosphere, the subroutine terminates and returns a flag to the main program.

    A complete description of the mathematical formulation of this subroutine can be found in Appendix D.

18. For each type of entry for a conservatively large W/C_DA a function has been defined for $\dot{\theta}_o$ as a function of L/D where $\dot{\theta}_o$ is that value of $\dot{\theta}$ at which the effective lift on the body becomes zero.

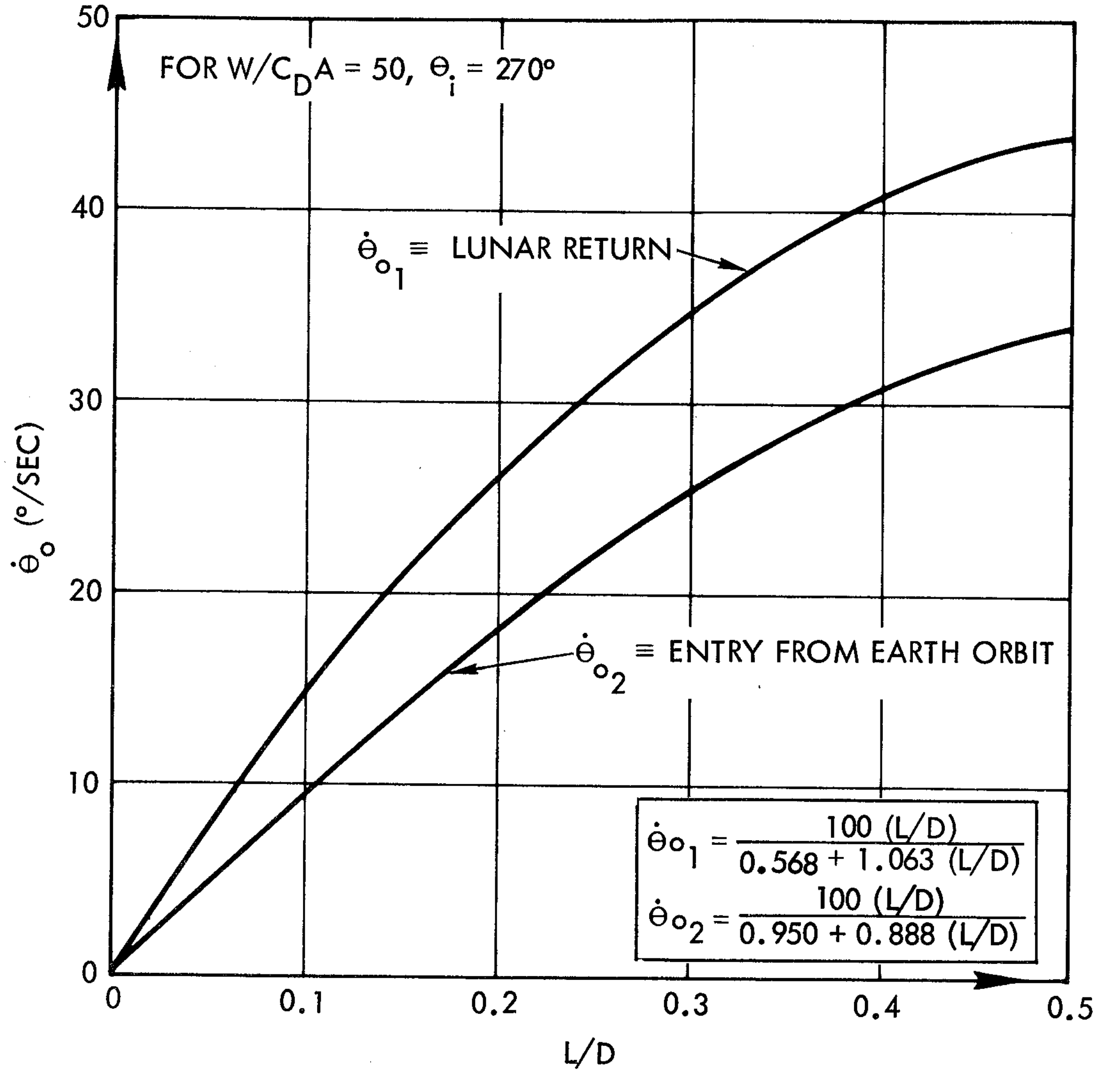


Figure 4. Critical Bank Angle Rates

Prior to each step along the reentry trajectory the subroutine computes a $\dot{\theta}$. This value is then compared to the appropriate function above. If at any time the computed $\dot{\theta} \geq \dot{\theta}_o$, L/D will be set equal to 0 for the remainder of that reentry.

When the program begins execution, an internal initialization is performed. This is represented by the broken lines on the flow diagram and takes place in the next two elements.

19. The output will be computed about some reference point on the surface of the earth. Typically this point represents the impact location for the vehicle if it had remained whole. The program computes this reference point based on the given reference ballistic coefficient from element 16.

20. The next initialization is an empirical set-up for pieces for which the effective L/D = 0. This is done in order to bypass a complete reentry computation whenever possible in order to minimize computing time.

    The relation between miss distance relative to the reference point and ballistic coefficient for non-lifting pieces is given by

$$DR = K_1 (W/C_D A)^{K_2}$$

$$CR = 0$$

    where

    DR = downrange miss (uprange, if negative), n mi

    CR = crossrange miss, n mi

    K_1 and K_2 = constants

    The program computes the relative miss distances for several ballistic coefficients over a range of 1 to 100 based on the initial conditions specified in element 16. A log-log table is then generated for DR versus $W/C_D A$ which by interpolation gives the proper relationship between any arbitrary $W/C_D A$ and its corresponding miss relative to the reference point for the given initial state vector of that mission.

21. Based on the decision made in element 13, certain cycles through the program will bypass the reentry subroutine. For those cycles the effective L/D = 0 and the miss distances are found from the table in element 20.

22. For those cycles which call on the reentry subroutine, it computes a downrange (DR) and crossrange (CR) miss relative to the reference point.

23. The program stores DR and CR or the occurrence of a skip for that cycle.

24. Input to the program the confidence and the maximum number of cycles desired for the particular piece.

25. The program executes a minimum of 200 cycles and then computes a downrange standard deviation, σ_{DR}. The program then computes the number of cycles which are required to estimate the σ_{DR} within 10 n mi of its true value with the given confidence as determined by the graph in Figure 5 (Reference 4). The program proceeds through this number of cycles unless it is greater than the maximum number of cycles allowed as stated in element 24.

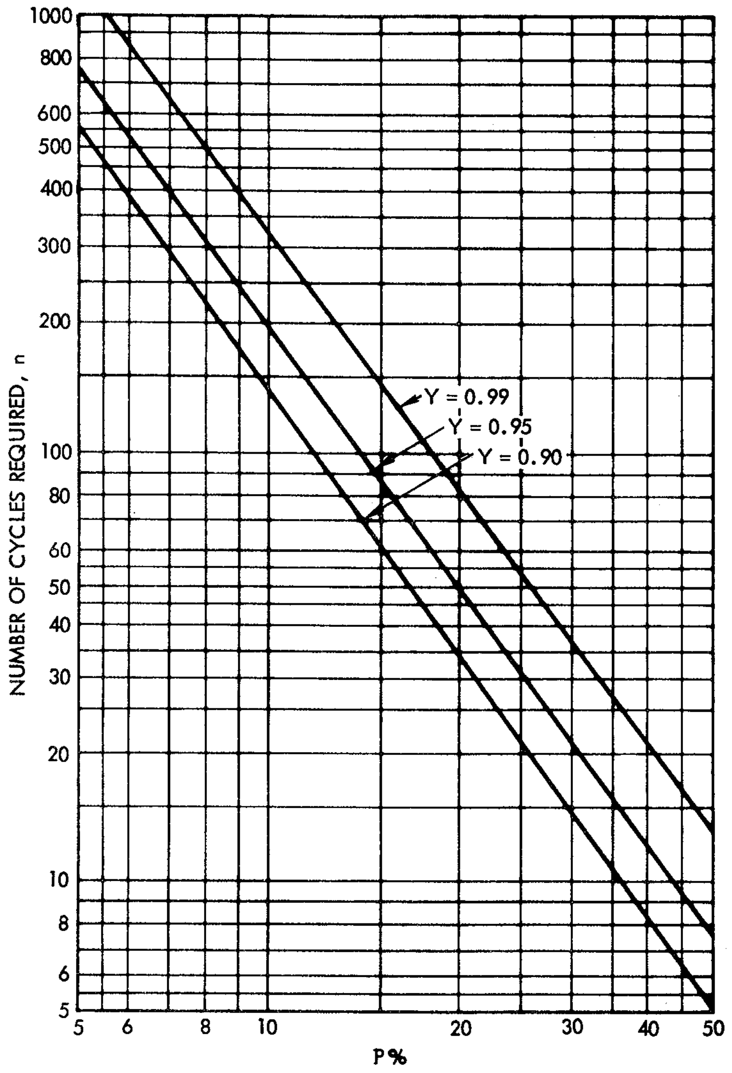


Figure 5. Number of Cycles Required to Estimate the Standard Deviation within P% of Its True Value (Where P% = $1000/\sigma$) With Confidence Coefficient Y

26. Once the program has completed the required number of cycles, it computes the output parameters. These output data will include the mean value of the downrange miss and crossrange miss relative to the breakup point, the standard deviation for the downrange and the crossrange miss relative to the breakup point. The covariance coefficient, the percent of the time that a skip condition occurred, the skew and kurtosis of the distribution, and the uprange and downrange standard deviation for a split bivariate normal approximation to the distribution. For the derivations and equations used in computing these quantities see Appendix E.

The program has been written in FORTRAN IV. A complete listing appears in Appendix A and an example of the input-output procedure is presented in Appendix B.

During this study, the program has been run on a CDC 6500 computer. The reentry subroutine determines a piece impact location and performs a miss distance computation in less than 1 second/cycle while the complete program computes the impact dispersion for a given piece (which is based on a randomly selected mix of lifting and effectively non-lifting reentries and a minimum of 200 cycles per piece) at an average rate of about 90 sec/piece. A typical mission (20 to 40 surviving pieces of debris) should therefore require less than 1 hour of central processor time.

4. CONCLUSIONS

The Monte Carlo simulation used in this study demonstrates that aerodynamic lift and other piece parameters can be treated in a rational fashion leading to a realistic impact distribution, rather than simply assuming that a worst case combination of parameters gives a three-sigma value for a normal distribution. Impact dispersions generated in this fashion are typically two orders of magnitude (10^2) smaller than those predicted by a worst case approach. Therefore, these results represent a far more realistic indication of the actual probability of impact in the vicinity of the nominal impact point, yet without excluding the possibility of impact at a remote location.

The program developed in this study offers a number of significant improvements over previous worst case techniques used to evaluate the impact distributions for pieces of reentering debris. For example:

- The state of a particular piece during reentry is randomly selected rather than simply assuming a worst case condition.
- The selection of the lift/drag ratio is based on a statistical distribution rather than some fixed value.
- A piece has a range of zero lift impact points based on a range of possible ballistic coefficients corresponding to different possible states rather than one single zero lift impact point.
- Lift vector orientation and bank angle rate have been included in the piece motion.
- The impact distribution is defined by a large number of cases and the accuracy of the output may be specified to any desired confidence.
- The output not only gives the mean and standard deviation in the downrange and crossrange directions but also the skew and the kurtosis of the distribution as well as the downrange standard deviation, the uprange standard deviation, and the point of split for the best fit of a split bivariate normal distribution to the computed data.

In addition, the program presented in this report offers a number of significant improvements over the parametric-empirical Monte Carlo simulation presented in Reference 1. For example:

- The program is highly flexible. It can be used for any mission, given the complete state vector at breakup and a list of surviving debris.
- The program provides a fast response time. There is no longer any need for a parametric study or program modification prior to its use for a specific mission.
- Accuracy has been improved since the use of "best fit" functions for parametric data has been eliminated.
- The program will provide a significant savings in the cost of debris analysis on an intermediate or long term basis. The reentry computations have been formulated to execute very rapidly and the use of an empirical table (which the program generates itself) allows the bypassing of the reentry routine for non-lifting cases. While the cost of computer running time involved in a mission analysis is several times greater than that required for a completely empirical program, a substantial saving is realized when one considers the several hours of computer time and many man-hours required to perform a parametric study.

5. REFERENCES

1. Marx, M., "Apollo Forced Entry Debris Dispersion Study," TRW Note 68-FMT-648 (5952-H492-R0-00), 25 April 1968.

2. Marx, M., "Apollo Forced Entry Debris Dispersion Study (U)," Addendum, TRW Note 68-FMT-649 (05952-H492-R8-01), 25 April 1968, Confidential.

3. Wilcox, L., "Task MSC/TRW A-116, Distribution of Impact Probability for Lifting Fragments," TRW IOC 67.6520.05-140, 26 October 1967.

4. Matrella, M. G., Experimental Statistics, National Bureau of Standards Handbook 91, 1963.

6. BIBLIOGRAPHY

Burington and May, Handbook of Probability and Statistics with Tables, Handbook Publications, Inc., 1958.

Hahn, G. J. and Shapiro, S. S., Statistical Models in Engineering, Wiley and Sons, Inc., 1967.

Lippman, G. W., "Aerodynamic Analysis of the Apollo C Service Module Debris for a Forced Entry," IOC 68-3324.3-118, 2 July 1968.

Palmiter, M., "Improved Routine for Rapid Reentry Calculation," IOC 3412.8-138, 23 August 1968.

APPENDIX A

PROGRAM LISTING

```
 PROGRAM MAIN(INPUT,OUTPUT,TAPE2=INPUT,TAPE3=OUTPUT)
 EXTERNAL FCOMP
 COMMON /INP/ H,VI,GAM,BLAT,BLON,AZI,LSD,WCDA,
1THETA,TDI,CONT,TTHDOT,INIT,SK,DR,CR,RFLAT,RFLON
 COMMON /SOX/FLG1,FLG2,C1,C2,C3,C4,C5,C6,A,B,C,IDER
 COMMON /STATIS/ REFLAT,REFLON,DRBAR,CRBAR,SIGDR,
1SIGCR,COVDC,RHO,PSKIP,NITER,EX3,EX4,SIGURF,
2SIGDRF,XR
 COMMON /STUFF/ COMNLD,MAXLD,MINLD,SIGMA,DX,UB,EPS,G,X(3),IRT
 EQUIVALENCE (REFLAT,STATIS(1))
 EQUIVALENCE (CD0,CDO)
 EQUIVALENCE (Y,P)
 DIMENSION HEAD(15),STATIS(15)
 DIMENSION AX(3), AH(3), BOUNDS(3), XBNDS(3), BLOCK(75), AF(3),
* JVECTR(3)
 DIMENSION XXX(4000)
 DIMENSION CRT(2),AWCDA(3),ALSD(3),BETA(1001),PROB(1001)
 DIMENSION WCDTAB(5),TBWCD(6),TBDR(6),TBUR(6)
 REAL LSD,             NSKIP,NITER,NUMAX,MINLD,MAXLD,MEANLD
 INTEGER ENTRY
 INTEGER SK
 DATA AX/ 500.,50.,500./, AH/ 3*0./, IBNDS/ 0/, XBNDS/ 5.E3,5.E2,
* 5.E3/, MAXIT/ 200/, EPSLN/1.E-5/,IFIN/0/
 DATA JVECTR /0,0,0/
 DATA WCDTAB /1.0,3.0,10.0,30.0,100.0/
 DATA HEAD( 1) /6HREFLAT /, HEAD( 2) /6HREFLON /
1,     HEAD( 3) /6HDRBAR  /, HEAD( 4) /6HCRBAR  /
2,     HEAD( 5) /6HSIGDR  /, HEAD( 6) /6HSIGCR  /
3,     HEAD( 7) /6HCOVDC  /, HEAD( 8) /6HRHO    /
4,     HEAD( 9) /6HPSKIP  /, HEAD(10) /6HNITER  /
5,     HEAD(11) /6HEX3    /, HEAD(12) /6HEX4    /
6,     HEAD(13) /6HSIGURF /, HEAD(14) /6HSIGDRF /
7,     HEAD(15) /6HXR     /
 NAMELIST /AR050/ MINLD,COMNLD,MAXLD,CDO,CDT,
1CDMAXL,H,VI,GAM,BLAT,BLON,AZI,WCDARF,AREF,ALPHA,FI,W,
2 TDI,Y,NUMAX,SIGMA,ERR,ENTRY
 MINLD=0.0
 COMNLD=0.0
 MAXLD=0.0
 CDO=0.0
 CDT=0.0
 CDMAXL=0.0
 H=0.0
 VI=0.0
 GAM=0.0
 BLAT=0.0
 BLON=0.0
 AZI=0.0
 THETA=0.0
 THDOT=0.0
 WCDARF=0.0
 AREF=0.0
 FI=0.0
 W=0.0
```

```
      NUMAX=0.0
      ENTRY=1
      NUMUR=1
      NUMNR=1
      SPI=1.7724539
      C1= SPI/SQRT(2.)
      C2= 1.5*C1
      C3= SQRT(8.)/SPI
      C4= -C3
      C5= 1.5
      C6= SQRT(2.)*SPI
      FLG1=0.0
      FLG2=0.0
      ERR=0.0
      DX=0.0
      DO 101 I=1,101
      PROB(I)=DX
      DX=DX+0.01
  101 CONTINUE
      EPS=0.0
      DX=0.01
      UB=10.0
      IDER=0
      DO 700 IA=1,3
  700 BOUNDS(IA)=XBNDS(IA)
C         RETURN FOR NEW CASE
 100  CONTINUE
      ALPHA=0.010
      TDI=0.0
      SIGMA=0.0
      READ (2,AR050)
      WRITE (3,AR050)
      FNUOLD=200.0
      NSKIP=0.0
      TITER=0.0
      NITER=0.0
      DRSUM=0.0
      CRSUM=0.0
      DRCRS=0.0
      DR2S=0.0
      CR2S=0.0
      C=0.0
C              GENERATE A BETA DISTRIBUTION
      ERRS=ERR
      CALL BETAD(PROB,BETA,101,ERR)
      IF(ERRS.NE.ERR) CALL EXIT
      IF(ERR.EQ.4.0) WRITE(3,1001)
 1001 FORMAT (1H1,4X,3HL/D,8X,5HW/CDA,7X,5HTHETA,9X,2HDR,10X,2HCR)
      DMMLD=MAXLD-MINLD
      DO 1 I=1,101
    1 BETA(I)=BETA(I)*DMMLD+MINLD
C              CONSTANT FOR THETA DOT EQUATION
      CTD=.0086740*W*VI*VI/FI*EXP(-H/23500.0)
      IF (ENTRY.EQ.2) CTD=10.0*CTD
```

```
C         COMPUTE REFERENCE IMPACT POSITION VECTOR
      INIT=0
      WCDA=WCDARF
      LSD=0.0
      CALL TRAJ
      REFLAT=RFLAT
      REFLON=RFLON
C         SETUP TABLE FOR L/D=0. IMPACT POINTS
      DO 110 I=1,5
      WCDA=WCDTAB(I)
      CALL TRAJ
      TBWCD(I)=ALOG(WCDA)
      TEMP=ALOG(ABS(DR))
      IF (DR.LT.0.)GO TO 109
      TBDR(I)=TEMP
      TBUR(I)=0.
      GO TO 110
  109 TBDR(I)=0.
      TBUR(I)=TEMP
  110 CONTINUE
      TEMP=ALOG(WCDARF)
      DO 111 I=1,5
      J=7-I
      K=J-1
      IF (TEMP.GE.TBWCD(K)) GO TO 112
      TBWCD(J)=TBWCD(K)
      TBDR(J)=TBDR(K)
  111 TBUR(J)=TBUR(K)
      J=1
  112 TBWCD(J)=TEMP
      TBDR(J)=1.0E-6
      TBUR(J)=1.0E-6
C****
C****
C                    S T A R T   I T E R A T I O N
C****
C              PICK L/D UNIFORMILY FROM BETA DISTRIBUTION
  200 CALL RDM1(NUMUR,RN)
      I=99.0*RN+2.0
      LSD=BETA(I)
C         COMPUTE C SUB D
      IF (LSD-COMNLD) 210,215,220
  210 CD=(CDMAXL-CDT)/(MAXLD-COMNLD)*(LSD-COMNLD)+CDT
      GO TO 225
  215 CD=CDT
      GO TO 225
  220 CD=(CDT-CDO)/(COMNLD-MINLD)*(LSD-MINLD)+CDO
C         COMPUTE W/C(D) A
  225 WCDA=W/(CD*AREF)
C              PICK THETA DOT AS FOLLOWS-
C                    USE RAYLEIGH DISTRIBUTION AND ALPHA INPUT TO FIND EL
C                    CALCULATE VALUE FROM FUNCTION(RN2S IS NORMAL RANDOM)
      CALL RN2S(RN1,NUMNR)
      CALL RN2S(RN2,NUMNR)
      EL=ALPHA*SQRT(RN1**2+RN2**2)
      DOT=CTD*EL*LSD/WCDA
C        TEST TO INTEGRATE TRAJECTORY OR USE TABLES
C          TO FIND DR AND CR
      IF (DOT.GE.1.0) GO TO 701
C        TRAJECTORY WILL BE INTEGRATED
C             GENERATE TEST THETA DOT
      IF (ENTRY.EQ.2) GO TO 226
      TTHDOT=.5680+1.0630*LSD
      GO TO 228
 226  TTHDOT=.9500+.8880*LSD
 228  TTHDOT=100.0*LSD/TTHDOT
C        PICK THETA (0 - 360 DEGREES)
      CALL RDM1(NUMUR,RN)
      THETA=RN*360.0
      CONT=EL*W/(FI*50.57964)  *57.29578
      SLSD=LSD
C        TRAJ INTEGRATES TRAJECTORY AND COMPUTES DR AND CR
      CALL TRAJ
      LSD=SLSD
      IF (SK.EQ.0) GO TO 800
C        SKIP CASE
      IF (ERR.NE.4.0) GO TO 90
      WRITE (3,9000) LSD,WCDA,THETA
 9000 FORMAT (1X,F7.2,1X,2F12.2,14X,4HSKIP)
   90 NSKIP=NSKIP+1.0
      GO TO 91
C        L/D=0.  FIND DR FROM TABLE
 701  CONTINUE
      THETA=0.
      TEMP=ALOG(WCDA)
      DO 705 I=2,5
 705  IF (TEMP.LE.TBWCD(I)) GO TO 706
      I=6
 706  J=I-1
      K=I
      IF (WCDA.LT.WCDARF) GO TO 710
      TEMPA=(TBDR(K)-TBDR(J))/(TBWCD(K)-TBWCD(J))*
     1 (TEMP-TBWCD(J))+TBDR(J)
      DR=EXP(TEMPA)
      CR=0.0
      GO TO 800
 710  TEMPA=(TBUR(K)-TBUR(J))/(TBWCD(K)-TBWCD(J))*
     1 (TEMP-TBWCD(J))+TBUR(J)
      DR=-EXP(TEMPA)
      CR=0.0
C        NON SKIP CASE
 800  CONTINUE
C             SAVE VALUES FOR OUTPUT
      NITER=NITER+1.0
      NNN=NITER
      XXX(NNN)=DR
      DRSUM=DRSUM+DR
      CRSUM=CRSUM+CR
      DRCRS=DRCRS+DR*CR
      DR2S=DR2S+DR*DR
      CR2S=CR2S+CR*CR
      C=C+DR*DR*DR
      IF (ERR.NE.4.0) GO TO 91
      WRITE (3,5000) LSD,WCDA,THETA,DR,CR
 5000 FORMAT (1X,F7.2,1X,2F12.2,2X,2F11.2)
   91 TITER=TITER+1.0
C             CONVERGENCE TEST
   94 IF(TITER.LT.FNUOLD) GO TO 200
      PERCNT=1000./SQRT((DR2S-(DRSUM**2)/NITER)/(NITER-1.0))
      FNU=1050.*PERCNT**(P*1.104262-3.086601)/(.3749680
     1-P*.3445660 )
   98 FNUNEW=FNU+1.0
      IF(FNUNEW.LE.FNUOLD) GO TO 600
      IF(FNUOLD.GE. NUMAX) GO TO 500
      FNUOLD=FNUNEW
      IF(FNUOLD.GT. NUMAX) FNUOLD= NUMAX
      GO TO 200
  500 WRITE(3,1000)
 1000 FORMAT(25H1PROCESS DID NOT CONVERGE)
  600 DRBAR=DRSUM/NITER
      CRBAR=CRSUM/NITER
      SIGDR=SQRT((DR2S-NITER*DRBAR**2)/(NITER-1.0))
      SIGCR=SQRT((CR2S-NITER*CRBAR**2)/(NITER-1.0))
      COVDC=(DRCRS-NITER*DRBAR*CRBAR)/(NITER-1.0)
      RHO=COVDC/(SIGDR*SIGCR)
      PSKIP=NSKIP/TITER
      A=DRBAR
      B=DR2S/NITER
      C=C/NITER
      EX3=0.0
      EX4=0.0
      DO 601 I=1,NNN
      TEMP=XXX(I)-DRBAR
      TEMPA=TEMP*TEMP
      EX3=EX3+TEMP*TEMPA
      EX4=EX4+TEMPA*TEMPA
  601 CONTINUE
      EX3=EX3/NITER
      EX4=EX4/NITER
      AX(1)=SIGDR
      AX(2)=SIGDR
      AX(3)=DRBAR
      IBNDS=1
      CALL BEGS
     * (AX,AH,IBNDS,BOUNDS,3,3,MAXIT,IDER,IFIN,JVECTR,EPSLN,
     * BLOCK,FCOMP,ITR,AF,FNORM)
      SIGDRF=AX(1)
      SIGURF=AX(2)
      XR=AX(3)
      WRITE (3,9001) (HEAD(I),STATIS(I),I=1,15)
 9001 FORMAT (1H1,15(/10X,A6,3H = ,F15.2))
      GO TO 100
```

END

```
      SUBROUTINE TRAJ
      COMMON /ACC / ACC(18)
      COMMON /CMIS/ CMIS(20)
      COMMON /CONT/ R,V,VA,CJD,BETA,ALT,ETAP,G
      COMMON /CONV/ CONV(30)
      COMMON /GRAV/ GRAV(12)
      COMMON /INP/ H,VI,GAMI,LATI,LONI,AZI,
     1LSD,WCDA,THETA,TDI,CONT,TTHDOT,INIT,SK,DR,CR
     2,RFLAT,RFLON
      COMMON /ITER/ ITER(6)
      COMMON /OUT/ OUT(15)
      COMMON /STAT/ STAT(18)
      COMMON /TIME/ TT,DT,TMAX,TMIN
      COMMON /VECT/ VECT(18)
      EQUIVALENCE (FVX,ITER(1)),(FOX,ITER(4))
      EQUIVALENCE (DOX,GRAV(1)),(LOX,GRAV(4))
      EQUIVALENCE (GOX,GRAV(7)),(URX,GRAV(10))
      EQUIVALENCE (ALP,ACC(1)),(BTP,ACC(4))
      EQUIVALENCE (GMP,ACC(7)),(AL,ACC(10))
      EQUIVALENCE (BT,ACC(13)),(GM,ACC(16))
      EQUIVALENCE (BE,CONV(1)),(EB,CONV(10))
      EQUIVALENCE (AA,CONV(19)),(BB,CONV(22))
      EQUIVALENCE (CC,CONV(25)),(AAA,CONV(28))
      EQUIVALENCE (VIXP,STAT(1)),(ROX,STAT(4))
      EQUIVALENCE (VOX,STAT(7)),(VAX,STAT(10))
      EQUIVALENCE (VEX,STAT(13)),(CNX,STAT(16))
      EQUIVALENCE (P,VECT(1)),(E2X,VECT(4))
      EQUIVALENCE (U1X, VECT(10))
      EQUIVALENCE (ROOX,OUT(7)),(VOOX,OUT(10))
      EQUIVALENCE (REFX,OUT(13))
      EQUIVALENCE (CK1,OUT(1)),(CK2,OUT(4))
      EQUIVALENCE (SRO,CMIS(1)),(SVO,CMIS(4))
      EQUIVALENCE (RIX,CMIS(7)),(VIX,CMIS(10))
      EQUIVALENCE (TA,CMIS(13)),(HA,CMIS(17))
      EQUIVALENCE (UIX,STAT(1))
      EQUIVALENCE (VEY,VEX(2)),(VEZ,VEX(3))
      EQUIVALENCE (CNY,CNX(2)),(CNZ,CNX(3))
      EQUIVALENCE (ROY,ROX(2)),(ROZ,ROX(3))
      EQUIVALENCE (VOY,VOX(2)),(VOZ,VOX(3))
      EQUIVALENCE (VAY,VAX(2)),(VAZ,VAX(3))
      EQUIVALENCE(TB ,TA (2)),(TC ,TA (3))
      EQUIVALENCE (TD ,TA (4)),(HB ,HA (2))
      EQUIVALENCE (HC ,HA (3)),(HD ,HA    (4))
      EQUIVALENCE (RIY,RIX(2)),(RIZ,RIX(3))
      DIMENSION DOX(3),GOX(3),URX(3)
      DIMENSION ALP(3),BTP(3),GMP(3),AL(3),BT(3),GM(3)
      DIMENSION BE(3,3),EB(3,3),AA(3),BB(3),CC(3)
      DIMENSION VIXP(3),ROX(3),VOX(3),VAX(3),VEX(3),CNX(3)
      DIMENSION P(3),E2X(3),E3X(3),U1X(3),U2X(3),U3X(3)
      DIMENSION RIX(3),VIX(3),ROOX(3),VOOX(3),REFX(3),
     1  CK1(3),CK2(3),UIX(3),TA(4),HA(4),HB(2)
      DIMENSION TEMP(6),TEM(6),TB(2)
      DIMENSION SRO(3),SVO(3)
      DIMENSION FVX(3),FOX(3),AAA(3)
      DIMENSION UNX(3)
      DIMENSION SSRO(3),SSVO(3)
      REAL LOX(3)
      REAL LATI,LONI,LAT,LON,LSD
      REAL NMFT
      INTEGER SK
      DATA DEG /57.2957795/
      DATA E1 /0.10/
      DATA E2 /0.250/
      DATA GSL /32.08763/
      DATA NMFT/1.6447E-4/
      DATA  PI/ 3.14159265/
      DATA RE /2.092460E7/
      DATA  WE/ 7.27220E-5/
      IF (INIT.NE.0) GO TO 20
      BETA=1.0/2.350E4
      LAT=LATI/DEG
      LON=LONI/DEG
      GAM=GAMI/DEG
      AZ=AZI/DEG
      T001=COS(LAT)
      T002=SIN(LAT)
      T003=COS(LON)
      T004=SIN(LON)
      T005=COS(GAM)
      T006=SIN(GAM)
      T007=COS(AZ)
      T008=SIN(AZ)
C        FIND INERTIAL COORDINATES
      TEMP(1)=T001*T003
      TEMP(2)=T001*T004
      TEMP(3)=T002
      TEMP(6)=T006*T001-T002*T005*T007
      TEMP(4)=TEMP(6)*T003-T004*T008*T005
      TEMP(5)=TEMP(6)*T004+T003*T008*T005
      TEMP(6)=T002*T006+T001*T005*T007
      T=RE+H
C        ROOX(I) IS POSITION VECTOR AT BREAKUP
C        VOOX(I) IS VELOCITY VECTOR AT BREAKUP
      DO 10 I=1,3
      ROOX(I)=T*TEMP(I)
      VOOX(I)=VI*TEMP(I+3)
      ROX(I)=ROOX(I)
      U1X(I)=VOOX(I)/VI
      URX(I)=ROOX(I)/T
   10 VOX(I)=VOOX(I)
      CALL VCTCRP (URX,U1X,UNX)
      HSKIP=H+50000.0
      DTEST=COS(1.0/DEG)
      GO TO 22
   20 DO 21 I=1,3
      ROX(I)=ROOX(I)
   21 VOX(I)=VOOX(I)
   22 SK=0
```

```
      FLAG=0.0
      ALT=H
      THDOT=TDI
      IF (ABS(THDOT).LT.1.0E-6) THDOT=1.0E-6
      DT=4.0
      TT=0.0
      TMIN=1.0
      TMAX=90.0/THDOT
      IF (TMAX.GT.16.0) TMAX=16.0
C        START INTEGRATION LOOP
 50   CONTINUE
      DO 501 I=1,3
      SRO(I)=ROX(I)
 501  SVO(I)=VOX(I)
      STT=TT
      SALT=ALT
 502  CONTINUE
      VAX= VOX+WE*ROY
      VAY=VOY-WE*ROX
      VAZ=VOZ
      CALL VCTMAG(VAX,VA)
      CALL VCTMAG(ROX,R)
      DO 52 I=1,3
      U1X(I)=VAX(I)/VA
 52   URX(I)=ROX(I)/R
      CALL VCTCRP (U1X,UNX,U3X)
      T=DOT (U1X,UNX)
      T=1.0/SQRT(1.0-T*T)
      DO 54 I=1,3
 54   U3X(I)=U3X(I)*T
      CALL VCTCRP (U3X,U1X,U2X)
C          J SUB D
      CJD=EXP(-BETA*(ALT-2.0E5))*.9840E-5/WCDA
      CNY=CJD*VA
      CNZ=CNY
      CNX=2.0*CNY
      VEX=0.0
      VEY=0.0
      CALL VCTMAG (VOX,V)
      VEZ=-R*WE/V
C
      T1=-GSL*RE*RE/(R*R)
      T2=-CJD*VA
      G=-T1
C          GRAVITY ACCELERATION
C          DRAG ACCELERATION
      DO 62 I=1,3
      GOX(I)=URX(I)*T1
      DOX(I)=VAX(I)*T2
 62   CONTINUE
      IF (LSD.EQ.0.0 .OR. TT.EQ.0.0) GO TO 509
      THDOT=TDI+TT*CONT*CJD*LSD*VA*VA
      TMAX=90.0/THDOT
      IF (TMAX.GT.16.0) TMAX=16.0
      IF (THDOT.LT.TTHDOT) GO TO 509
      LSD=0.0
      TMAX=16.0
  509 CONTINUE
C         STEP SIZE SELECTION
C         IF FIRST STEP OF CASE, DO NOT EXECUTE
      IF (ALT.LT.100000.0) GO TO 550
      IF (FLAG.EQ.0.0) GO TO 550
      KKK=0
  510 T1=(THETA+THDOT*(TT+DT/2.0))/DEG
      CALL VCTMAG (LOX,LX)
      CALL VCTMAG (DOX,DX)
      T2=COS(T1)
      T1=SIN(T1)
      T4=THDOT*DT
      T3=CJD*VA*VA*LSD*SIN(T4/DEG)/T4
      DO 511 I=1,3
      T4=U2X(I)*T1+U3X(I)*T2
  511 TA(I)=DOX(I)+GOX(I)+(T3*T4)/V
      T1=TT+DT
      CALL PFUN(T1)
      CALL VFUN(T1)
      DO 514 I=1,3
      TEMP(I)=ROX(I)+R*P(I)
  514 TEMP(I+3)=VOX(I)+V*FVX(I)
      TEMP(4)=TEMP(4)+TEMP(2)*WE
      TEMP(5)=TEMP(5)-TEMP(1)*WE
      CALL VCTMAG (TEMP,T1)
      CALL VCTMAG (TEMP(4),T2)
      DO 516 I=1,3
      TEM(I)=TEMP(I)/T1
  516 TEM(I+3)=TEMP(I+3)/T2
      T3=EXP(-(T1-RE-2.0E5)*BETA)*.9840E-5/WCDA
      T5=-GSL*RE*RE/(T1*T1)
      T6=-T3*T2
      T7=T3*T2*T2*LSD*SIN(T4/DEG)/T4
      DO 517 I=1,3
  517 HA(I)=T5*TEM(I)+T6*TEMP(I+3)
      CALL VCTCRP (TEM(4),UNX,TEM(1))
      T2=DOT(TEM(4),UNX)
      T2=1.0/SQRT(1.0-T2*T2)
      DO 518 I=1,3
  518 TEM(I)=TEM(I)*T2
      CALL VCTCRP (TEM(1),TEM(4),TEMP(1))
      T4=(THETA+1.50*DT*THDOT)/DEG
      T3=SIN(T4)
      T4=COS(T4)
      DO 520 I=1,3
  520 HA(I)=TA(I)-HA(I)-T7*(TEMP(I)*T3+TEM(I)*T4)
      CALL VCTMAG (TA,T1)
      CALL VCTMAG (HA,T2)
C         T1 IS EVALUATION CRITERION
      T1=T2/T1
C         TEST FOR DOUBLING OF STEP SIZE
```

```
      IF (T1.GE.E1) GO TO 524
      DT=DT*2.0
      KKK=511
      IF (DT.LT.TMAX) GO TO 510
      DT=TMAX
      GO TO 550
C         TEST FOR HALVING OF STEP SIZE
 524  IF (T1.LT.E2) GO TO 550
      IF (DT.LE. TMIN) GO TO 550
      DT=DT/2.0
      IF (KKK.EQ.511) GO TO 550
      IF (DT.LT.TMIN) DT=TMIN
      FLAG=0.
      DO 525 I=1,3
      ROX(I)=SSRO(I)
 525  VOX(I)=SSVO(I)
      TT=SSTT
      ALT=SSALT
      GO TO 50
C         END OF STEP SIZE SELECTION
 550  CONTINUE
      FLAG=511.0
C         CONTINUE INITIALIZATION
      T=(THETA+THDOT*(TT+DT/2.0))/DEG
      T1=SIN(T)
      T2=COS(T)
      DO 56 I=1,3
 56   E2X(I)=U2X(I)*T1+U3X(I)*T2
      CALL VCTCRP(U1X,E2X,E3X)
      T= THDOT*DT/DEG
C         ETA PRIME
      ETAP= LSD*SIN(T)/T
C
C        ECI TO BCS  AND  BCS TO ECI MATRIX SETUP
      DO 58 I=1,3
      EB(1,I)=U1X(I)
      EB(2,I)=E2X(I)+U1X(I)*ETAP*2.0
      EB(3,I)=E3X(I)
      BE(I,1)=U1X(I)-E2X(I)*ETAP*2.0
      BE(I,2)=E2X(I)
 58   BE(I,3)=E3X(I)
      T=ETAP*CJD*VA*VA
C         LIFT ACCELERATION
      DO 64 I=1,3
      LOX(I)=E2X(I)*T
      AA(I)=GOX(I)+DOX(I)+LOX(I)
      BB(I)=VOX(I)/R
      CC(I)=AA(I)/(2.0*R)
C         A
      AAA(I)=AA(I)/V
 64   CONTINUE
      CALL FUN(BB,BB)
      CALL FUN(CC,CC)
C         A PRIME
```

```
      CALL MXMNMP (EB,AAA,AA)
C          B PRIME
      CALL MXMNMP (EB,BB,BB)
C          C PRIME
      CALL MXMNMP (EB,CC,CC)
C          ALPHA PRIME
C          BETA PRIME
C          GAMMA PRIME
      DO 66 I=1,3
      T1=CNX(I)*CNX(I)
      ALP(I)=AA(I)/CNX(I)-BB(I)/T1+(2.0*CC(I))/(T1*CNX(I))
      BTP(I)=BB(I)/CNX(I)-(2.0*CC(I))/T1
 66   GMP(I)=CC(I)/CNX(I)
C          ALPHA
      CALL MXMNMP (BE,ALP,AL)
C          BETA
      CALL MXMNMP (BE,BTP,BT)
C          GAMMA
      CALL MXMNMP (BE,GMP,GM)
C
      T1=TT+DT
C          P FUNCTION          EVALUATE P
      CALL PFUN(T1)
C          NU FUNCTION         EVALUATE FVX
      CALL VFUN(T1)
      DO 76 I=1,3
      RIX(I)= ROX(I)+R*P(I)
 76   VIX(I)= VOX(I)+V*FVX(I)
      CALL VCTMAG(RIX,HI)
C
      HI=HI-RE
      TT=T1
C         CHECK FOR SKIP
      IF(HI.GT.HSKIP) GO TO 500
C         CHECK FOR IMPACT
      IF (HI.LE.0.) GO TO 600
C         CHECK FOR DROP
      TA(1)=VIX+WE*RIY
      TA(2)=VIX(2)-WE*RIX
      TA(3)=VIX(3)
      CALL VCTMAG (TA,T2)
      IF ((ALT-HI)/(T2*DT).GT.DTEST) GO TO 560
C         NO DROP,
C         NO IMPACT,  UPDATE VECTORS
      DO 86 I=1,3
      SSVO(I)=SVO(I)
      SSRO(I)=SRO(I)
      ROX(I)=RIX(I)
 86   VOX(I)=VIX(I)
      SSTT=STT
      SSALT=SALT
      ALT=HI
C         CONTINUE INTEGRATION LOOP
      GO TO 50
```

```
C          SKIP CONDITION
  500   SK=1
        CR=0.
        DR=0.
        INIT=511
        RETURN
C          DROP
  560   CONTINUE
        T3=RE/(RE+HI)
        DO 552 I=1,3
  552   RIX(I)=RIX(I)*T3
        TI=TT
        GO TO 701
C          IMPACT
  600   TA=TT-DT
        TB=TA+DT/3.0
        TC=TB+DT/3.0
        TD=TT
        TT=TA
        HA=ALT
        HD=HI
        DO 601 I=1,2
        CALL PFUN(TB(I))
        DO 602 J=1,3
  602   RIX(J)=ROX(J)+R*P(J)
        CALL VCTMAG (RIX,HI)
  601   HB(I)=HI-RE
C          CUBIC INTERPOLATION FOR TIME WHEN H=0
        TI=0.0
        DO 650 I=1,4
        TEMP=1.0
        DO 640 J=1,4
        IF (I.EQ.J) GO TO 640
        TEMP=TEMP*HA(J)/(HA(I)-HA(J))
  640   CONTINUE
  650   TI=TI-TA(I)*TEMP
C          TI IS IMPACT TIME
        CALL PFUN(TI)
        DO 700 I=1,3
  700   RIX(I)=ROX(I)+R*P(I)
  701   CONTINUE
C          RIX IS STATE VECTOR AT IMPACT
        CALL VCTMAG (RIX,HI)
C          CALCULATE LATITUDE AND LONGITUDE OF IMPACT
        RFLAT=ASIN (RIZ/HI) *DEG
        RFLON=(ATAN2(RIY,RIX)-WE*TI)*DEG
C          IS THIS SETUP PASS
        IF (INIT.NE.0) GO TO 800
C           YES
        DR=0.0
        CR=0.0
C          SET REFERENCE STATE VECTOR
        DO 750 I=1,3
        REFX(I)=RIX(I)
```

```
      CK1(I)=RIX(I)/HI
  750 CK2(I)=REFX(I)-ROOX(I)
      CALL VCTCRP (ROOX,CK2,CK2)
      CALL VCTMAG (CK2,T1)
      DO 752 I=1,3
  752 CK2(I)=CK2(I)/T1
      INIT=511
      REFT=TI
      RETURN
C        CALCULATE DR=DOWNRANGE ERROR
C                 AND  CR=CROSSRANGE ERROR
  800 T1=-WE*(TI-REFT)/(2.0*DEG)
      T2=SIN(T1)
      T4=2.0*ABS(T1)
      T5=1.0-T2*T4
      T6=T4*COS(T1)
      UIX(1)=RIX*T5-RIY*T6
      UIX(2)=RIY*T5+RIX*T6
      UIX(3)=RIZ
      CALL VCTMAG (UIX,T3)
      DO 801 I=1,3
  801 UIX(I)=UIX(I)/T3
      T1=DOT (UIX,CK2)
      CR=ASIN(T1)*RE*NMFT
      T2=DOT(UIX,CK1)/SQRT(1.0-T1*T1)
      DR= ACOS(T2)*RE*NMFT
      CALL VCTCRP (CK2,CK1,TEMP)
      IF (DOT(UIX,TEMP).LT.0.0) DR=-DR
      INIT=511
      RETURN
      END
      SUBROUTINE VFUN(T)
C         NU FUNCTION
      COMMON /ACC/ ACC(1)
      COMMON /CONV/ BE
      COMMON /ITER/FVX
      COMMON /STAT/ STAT(1)
      COMMON /TIME/ TT
      EQUIVALENCE (CNX,STAT(16))
      EQUIVALENCE (ALP,ACC(1)),(AL,ACC(10))
      EQUIVALENCE (BT,ACC(13)),(GM,ACC(16))
      DIMENSION A(3),CNX(3),ALP(3),FVX(3)
      DIMENSION AL(3),BT(3),GM(3),BE(3,3)
      DT=T-TT
      DO 200 I=1,3
  200 A(I)=EXP(-CNX(I)*DT)*ALP(I)
      DO 220 I=1,3
      B=0.0
      DO 210 J=1,3
  210 B=B+BE(I,J)*A(J)
  220 FVX(I)=AL(I)+BT(I)*DT+GM(I)*DT*DT-B
      RETURN
      END
      SUBROUTINE PFUN(TI)
C          P FUNCTION
      COMMON /ACC/ ACC(1)
      COMMON /CONT/ R,V
      COMMON /CONV/ BE
      COMMON /STAT/ STAT(1)
      COMMON /TIME/ TT
      COMMON /VECT/ P
      EQUIVALENCE (ALP,ACC (1)),(AL ,ACC (10))
      EQUIVALENCE (BT ,ACC (13)),(GM ,ACC (16))
      EQUIVALENCE (VOX,STAT(7)),(CNX,STAT(16))
      DIMENSION A(3),CNX(3),ALP(3),AL(3),BT(3),GM(3)
      DIMENSION VOX(3),P(3),BE(3,3)
      T=TI-TT
      T2=T*T
      T3=T2*T
      DO 25 J=1,3
   25 A(J)=(EXP(-CNX(J)*T)-1.0)*ALP(J)/CNX(J)
      DO 30 I=1,3
      B=0.0
      DO 28 J=1,3
   28 B=BE(I,J)*A(J)+B
      B=(AL(I)+VOX(I)/V)*T+BT(I)*T2/2.0+
     1GM(I)*T3/3.0+B
   30 P(I)=V*B/R

      RETURN
      END
      SUBROUTINE FUN (QI,F)
C          F FUNCTION
      COMMON /CONT/ CONT(1)
      COMMON /CONV/ CONV(1)
      COMMON /GRAV/ GRAV(1)
      COMMON /STAT/ STAT(1)
      EQUIVALENCE (DOX,GRAV(1)),  (LOX,GRAV(4))
      EQUIVALENCE (GOX,GRAV(7)),  (URX,GRAV(10))
      EQUIVALENCE (BE ,CONV(1)),  (ROX,STAT(4))
      EQUIVALENCE (VAX,STAT(10)), (VEX,STAT(13))
      EQUIVALENCE (V  ,CONT(2)),  (VA ,CONT(3))
      EQUIVALENCE (CJD,CONT(4)),  (BETA,CONT(5))
      EQUIVALENCE (G  ,CONT(8))
      REAL LOX
      DIMENSION QI(3),Q(3),F(3),GOX(3),DOX(3),LOX(3)
      DIMENSION A(3),BE(3,3)
      Q(1)=QI(1)
      Q(2)=QI(2)
      Q(3)=QI(3)
      T1=DOT(URX,Q)*3.0/V
      T2=DOT(ROX,Q)*BETA/V
      CALL VCTCRP (VEX,Q,A)
      T3=DOT(VAX,A)*CJD
      DO 20 I=1,3
   20 F(I)=-Q(I)*G/V-GOX(I)*T1-(DOX(I)+LOX(I))*
     1T2-A(I)*CJD*VA-BE(I,1)*T3
      RETURN
      END
      FUNCTION DOT (A,B)
C        VECTOR INNER PRODUCT
      DIMENSION A(3),B(3)
      DOT=A(1)*B(1)+A(2)*B(2)+A(3)*B(3)
      RETURN
      END


      SUBROUTINE VCTCRP (A,B,D)
C        VECTOR CROSS PRODUCT
      DIMENSION D(3)
      DIMENSION A(3),B(3),C(3)
      C(1)=A(2)*B(3)-A(3)*B(2)
      C(2)=A(3)*B(1)-A(1)*B(3)
      C(3)=A(1)*B(2)-A(2)*B(1)
      D(1)=C(1)
      D(2)=C(2)
      D(3)=C(3)
      RETURN
      END


      SUBROUTINE VCTMAG (A,B)
C        VECTOR MAGNITUDE
      DIMENSION A(3)
      B=SQRT(DOT(A,A))
      RETURN
      END


      SUBROUTINE MXMNMP (A,B,C)
C        3X3 MATRIX AND 3X1 VECTOR MULTIPLICATION
      DIMENSION CC(3)
      DIMENSION A(3,3),B(3),C(3)
      DO 200 M=1,3
      DC=0.0
      DO 100 N=1,3
  100 DC=A(M,N)*B(N)+DC
  200 CC(M)=DC
      C(1)=CC(1)
      C(2)=CC(2)
      C(3)=CC(3)
      RETURN
      END
      SUBROUTINE BETAD (P,XT,NP,ERR)
      DIMENSION P(NP),XT(NP)
      COMMON /STUFF/ COMN,MAX,MIN,SS,DX,UB,EPS,G,X(3),IRT
      REAL MAX,MIN
      DATA C120 / 209.43950E-2/, C240/ 418.87900E-2/, RD/ 1.7453292E-2/
C                P - TABLE OF PROBABILITIES
C                XT- RESULTANT TABLE OF X'S
C                G - NU
C                S - SIGMA
      QUBE(X) = SIGN( ABS(X)**.33333333, X )
C
      G= COMN/(MAX-MIN)
      C1= (1.-G)/G
      XU=1.
      IF (SS.EQ. 0.) GO TO 220
      S=SS*SS
C
      C2=C1+1.
      P1=(7.0*S*C2**2-C1)/(S*C2**3)
      Q=(16.0*S-1.0)/(S*C2**2)
      R=(12.0*S-1.0)/(S*C2**3)
C
      SA = (3.*Q - P1**2)/3.
      SB = (2.*(P1**3) - 9.*P1*Q + 27.*R)/27.
      RAD= (SB**2) / 4. + (SA**3) / 27.
      IF ( ABS(SA-SB) .LT. EPS*(ABS(SA) + ABS(SB)) ) RAD=0.
      IF (RAD) 100,150,200
C                THREE REAL AND UNEQUAL ROOTS
  100 PHI = ACOS( (-SB/2.) / SQRT(-SA**3/27.) )/3.
      IRT=3
      T= 2.*SQRT(-SA/3.)
      X(1)= T*COS(PHI)
      X(2)= T*COS( PHI+C120 )
      X(3)= T*COS( PHI+C240 )
      GO TO 210
C                THREE REAL AND TWO EQUAL ROOTS
  150 CONTINUE
      IRT=2
      A= QUBE( -SB/2. )
      X(1)=2.*A
      X(2)=-A
      X(3)=X(2)
      GO TO 210
  200 CONTINUE
C                ONE REAL, TWO CONJUGATE COMPLEX ROOTS
      IRT=1
      A= QUBE( -SB/2. + SQRT(RAD) )
      B= QUBE( -SB/2. - SQRT(RAD) )
      XU=A+B
      X(1)=XU
      X(2)=XU
      X(3)=XU
  210 CONTINUE
```

```
C                   SET XU TO ROOT CLOSEST TO ZERO
      T=P1/3.
      DO 211 I=1,3
  211 X(I)=X(I)-T
      IF(X(1).LE.X(2)) GO TO 140
      XU=X(1)
      X(1)=X(2)
      X(2)=XU
  140 IF(X(2).LE.X(3)) GO TO 141
      XU=X(2)
      X(2)=X(3)
      X(3)=XU
  141 DO 142 I=1,3
      IF(X(I).LT.0.0) GO TO 142
      XU=X(I)
      GO TO 143
  142 CONTINUE
      ERR=1.0
      RETURN
  143 IF(ABS(XU).LE.UB) GO TO 220
      ERR=1.
      RETURN
  220 CONTINUE
      YU=XU*C1
C                   INTEGRATE F(X)
      XT(1)=0.
      XT(NP)= 1.
      A=XU+YU+2.0
      A=GAMMA(A)/(GAMMA(XU+1.0)*GAMMA(YU+1.0))
      XSP=0.
      XI=0.
      SUM=0.
      SUMP=0.
      KP=NP-1
      KKK=0
      KS=2
  310 CONTINUE
      XSP=XI
      XI=XI+DX
      IF (XI .LE. 1.) GO TO 320
      XI=1.
      KKK=KKK+1
      IF(KKK.LT.100) GO TO 320
      ERR=3.0
      GO TO 400
  320 CONTINUE
      SUMP=SUM
      SUM = SUM + (A * (XI**XU)*(1.-XI)**YU) * DX
  330 CONTINUE
      IF (SUM .LT. P(KS)) GO TO 310
C                   INTERPOLATE LINEARLY TO DETERMINE XT
  340 CONTINUE
      XT(KS) = ( P(KS)-SUMP ) * ( XI-XSP )/( SUM-SUMP) + XSP
      XTEST=XT(KS)
      PTEST=SUMP+(A*(XTEST**XU)*(1.0-XTEST)**YU)*DX
      IF(ABS(P(KS)-PTEST).LE.0.001) GO TO 341
      XI=XSP
      SUM=SUMP
      DX=0.5*DX
      GO TO 310
  341 KS=KS+1
      IF(KS.LE.KP) GO TO 330
  400 CONTINUE
      IF (ERR .EQ. 0.) RETURN
      WRITE(3,900) XU,YU
  900 FORMAT(1H1,10X,25HBETA DISTRIBUTION FOR L/D/ 1H0,
     14H A =F8.4,4H B =F8.4)
      WRITE (3,910) (KS,P(KS),XT(KS),KS=1,NP)
  910 FORMAT(1H , I4, 3H P=, F10.6,     3H X=, F10.6)
      RETURN
      END
      SUBROUTINE  RN2S(A,L)
C     RW  RN2S  NORMAL RANDOM NUMBER GENERATOR
C         THE PURPOSE OF THIS ROUTINE IS TO GENERATE INDEPENDENT RANDOM
C         NUMBERS, EACH NORMALLY DISTRIBUTED WITH MEAN ZERO AND STANDARD.
C         DEVIATION UNITY, IN FLOATING POINT FORM.
C            WHERE   A  IS THE LOCATION OF THE RESULTING FL-PT.
C                       RANDOM NUMBER.
C                    L  IS THE LOCATION OF A 12 DIGIT OCTAL NUMBER,
C                       INILIALLY A FIXED POINT OCTAL (1).
      EQUIVALENCE (AL,LL)
      DATA I/30517578125 /, J/0377777777777/
      DATA LL/1/, X2/0.0/
      IF ( L .EQ. 0) GO TO 10
      LL = L
   10 IF( X2 .EQ. 0.0) GO TO 20
      A  = X2
      X2 = 0.0
      RETURN
   20 ASSIGN 21 TO K
      GO TO 100
   21  U1 = U
      ASSIGN 22 TO K
      GO TO 100
   22 U2 = U
C     GENERATE  NORMAL RANDON NUMBER
      A1 = SQRT(-2.0 * ALOG(U1))
      A2 = 6.2831852 * U2
C     X2 = (-2.0 LOG(U1))**(1/2)  SIN ( 2 PI  U2)
      X2  = A1 * SIN(A2)
C     X1 = (-2.0 LOG(U1))**(1/2)  SIN ( 2 PI  U2)
      A   = A1 * COS(A2)
      L   =  LL
      RETURN
  100 LL = I* LL
C          THE FOLLOWING STATEMENT WAS MADE FOR COMPATIBILITY
      AL  = AND(LL,J)
      U = FLOAT(LL/256)/ 134217728.
      GO TO K,(21,22)
      END
      SUBROUTINE  RDM1(L,U)
C     RW RDM1 UNIFORM RANDOM NUMBERS
C     THE PURPOSE OF THIS ROUTINE IS TO GENERATE UNIFORMLY
C     DISTRIBUTED PSEUDO RANDOM NUMBERS.
C     WHERE LL IS THE LOCATION OF A 12 DIGIT OCTAL NUMBER
C              INITIALLY A FIXED POINT OCTAL (1).
C            U IS THE UNIFORMLY DISTRIBUTED PSEUDO RANDOM NUMBER
      EQUIVALENCE (AL,LL)
      DATA I/30517578125/, J/0377777777777/
      LL =L
      IF(LL .EQ. 0) LL =1
      LL = I* LL
      AL = AND(LL,J)
C     THE ABOVE STATEMENT WAS MADE FOR COMPATIBILITY
      U = FLOAT(LL/256)/ 134217728.
      L = LL
      RETURN
      END
      SUBROUTINE FCOMP (X,F,P,IBLEW)
      DIMENSION X(3),F(3),P(3,3)
      COMMON / SOX / FLG1,FLG2,C1,C2,C3,C4,C5,C6,A,B,C,IDER
      IBLEW=0
      F(1) = (X(1) - X(2) +C6*X(3)) - C6*A
      F(2) = (X(1)**2 + X(2)**2 + 2.*X(3)**2)
     *       + (C3*X(3)*X(1) + C4*X(3)*X(2)) - 2.*B
C
      F(3) = (X(1)**3 - X(2)**3 + C1*X(3)**3)
     *       + (C2*X(3)*X(1)**2 + C2*X(3)*X(2)**2)
     *       + (C5*(X(1)-X(2))*X(3)**2) - C1*C
      FN = F(1)**2 + F(2)**2 + F(3)**2
      IF (FLG1 .NE. 0.) WRITE(3,900) (X(I),I=1,3),(F(I),I=1,3),FN
  900 FORMAT(1H , 2HX=, 3E17.8, 3H F=, 3E17.8,8H NORM F=, E17.8)
      IF (IDER .EQ. 0) RETURN
      P(1,1) = 1.
      P(1,2) = -1.
      P(1,3) = C6
C
      P(2,1) = 2.*X(1) + C3*X(3)
      P(2,2) = 2.*X(2) + C4*X(3)
      P(2,3) = 4.*X(3) + C3*X(1) + C4*X(2)
C
      P(3,1) = 3.*X(1)**2 + 2.*C2*X(3)*X(1) + C5*X(3)**2
C
      P(3,2) = -3.*X(2)**2 + 2.*C2*X(3)*X(2) - C5*X(3)**2
C
      P(3,3) = 3.*C1*X(3)**2 + C2*X(1)**2 + C2*X(2)**2
     *            +2.*C5*(X(1)-X(2))*X(3)
      IF (FLG2 .NE. 0.) WRITE(3,901) ((P(I,J),J=1,3),I=1,3)
  901 FORMAT(1H ,2HP= ,   3E17.8/ (3X,3E17.8) )
      RETURN
      END
      FUNCTION GAMMA(XX)
      IF (XX - 57.0) 6,6,4
    4 WRITE (3,1)
    1 FORMAT (40H1 ERROR IN GAMMA FUNCTION - SIGNING OFF )
      STOP
    6 X = XX
      ERR = 1.0E-6
      IER = 0
      GX = 1.0
      IF (X - 2.0) 50,50,15
   10 IF (X - 2.0) 110,110,15
   15 X  = X - 1.0
      GX = GX * X
      GO TO 10
   50 IF (X - 1.0) 60,120,110
   60 IF (X - ERR) 62,62,80
   62 Y = FLOAT(INT(X)) - X
      IF (ABS(Y) - ERR) 130,130,64
   64 IF (1.0 - Y - ERR) 130,130,70
   70 IF (X - 1.0) 80,80,110
   80 GX = GX / X
      X=X+1.0
      GO TO 70
  110 Y = X - 1.0
      GY = 1.0 + Y * (-0.5771017 + Y * (+0.9858540 + Y * (-0.8764218 + Y
     1* (+ 0.83282124 + Y * (- 0.5684729 + Y * (+0.2548205 + Y * (-0.051
     249930)))))))
      GAMMA = GX * GY
  120 RETURN
  130 GO TO 4
      END
      SUBROUTINE BEGS(X,H,IBNDS,BNDS,MX,NY,
     1 MAXIT,IDEV,IFIN,IV,EPS,S,FCALC,NIT,F,BSST)
      DIMENSION S(1),X(1),H(1),BNDS(1),IV(1),F(1)
C
      AB(P)=ABS(P)
 9005 FORMAT(20H  ***SUM OF F(I)**2=E16.7)
 9010 FORMAT(7H0    X=7E16.7/(7X,7E16.7))
 9015 FORMAT(7H     F=7E16.7/(7X,7E16.7))
 9020 FORMAT(20H     SUM OF F(I)**2=E16.7)
 9025 FORMAT ( 4H0 X(I2,15H) PERTURBED BY E11.4)
C  TO CONVERT TO DOUBLE PRECISION, REMOVE THE
C  ABOVE 6 CARDS AND REMOVE THE C FROM
C  COLUMN 1 OF THE FOLLOWING 8 CARDS
C     DOUBLE PRECISION X,H,BNDS,S,F,BSST,BST,BF,
C    1 P,SF,A1,T,SUS,SUP
C     AB(P)=DABS(P)
C9005 FORMAT(20H  ***SUM OF F(I)**2=D27.16)
C9010 FORMAT(7H0    X=3D27.16/(7X,3D27.16))
C9015 FORMAT(7H     F=3D27.16/(7X,3D27.16))
C9020 FORMAT(20H     SUM OF F(I)**2=D27.16)
C9025 FORMAT(4H0 X(I2,15H) PERTURBED BY D27.16)
C
C RENAME INPUT ITEMS MX,NY,IDEV.
      M=MX
      N=NY
      EPSL=ABS(EPS)
      IDER=IDEV
C TEST IF TOLERANCE,EPSL, IS ZERO.
      IF(EPSL)20,10,20
C SUBSTITUTE REASONABLE VALUE FOR EPSL.
   10 EPSL=1.E-5
C INITIALIZE ITERATION COUNTER,NIT = 0. INITIALIZE NO. OF COMPONENTS
C TO BE REGARDED IN PERTURBATION OPERATION, IP = 0.
   20 NIT=0
      IP=0
C EVALUATE FUNCTION OF GUESS POINT, F(X).
      CALL FCALC(X,F,S(1),IBLEW)
C TEST IF GUESS POINT GOOD.
      IF(IBLEW)670,40,670
C COMPUTE NORM SQUARED,BF, OF FUNCTION AT GUESS POINT.
   40 BF=0.
      DO 50 I=1,N
   50 BF=BF+F(I)**2
C INITIALIZE BEST VALUE OF NORM SQUARED,BST.
      BST=BF
C TEST IF FINISHING PROCEDURE ONLY IS TO BE USED.
      IF(IFIN)900,60,900
   60 IF(BST.EQ.0.) GO TO 670
C ESTABLISH CONSTRAINT AND PERTURBATION VECTORS ACCORDING TO VECTOR
C INTEGER CODES ,IV(I).
      DO 150 I=1,M
C   TEST VECTOR OF INTEGERS FOR NON-ZERO COMPONENTS,I.E. DISREGARD
C   CORRESPONDING COMPONENTS OF X IN ITERATION PROCEDURE
```

```
      IF(IV(I))150,70,150
   70 IP=IP+1
C    TEST VECTOR OF CONSTRAINTS FOR ZERO COMPONENTS.
      IF(BNDS(I))100,90,100
C    RESET I-TH COMPONENT IF INITIALLY ZERO.
   90 BNDS(IP)=10.0
      GO TO 110
  100 BNDS(IP)=BNDS(I)
C    TEST FOR NULL COORDINATE IN PERTURBATION VECTOR-IF ZERO THEN RESET
  110 IF(H(I))150,120,150
  120 IF(X(I))130,140,130
  130 H(I)=.01*X(I)
      GO TO 150
  140 H(I)=.001
  150 CONTINUE
C DEFINE INDEX VALUES FOR TEMPORARY STORAGE ARRAY S.
      JJ=IP*IP
      JPFF=N*IP
      JPX=JPFF+N
      JY=JPX+JJ
      JAZ=JY+IP
      JW=JAZ+IP
      JQ=JW+IP
      JATA=JQ+JJ
      JAR=JATA+((IP+1)*(IP+2))/2
      JV=JAR+((IP+2)*(IP+3))/2
C INITIALIZE BEST F IN S ARRAY-- S(MN+1),....,S(MN+N)
      DO 170 I=1,N
      JJ=JPFF+I
  170 S(JJ)=-F(I)
C TEST IF USING TANGENT OR SECANT METHOD.
      IF(IDER)510,200,510
C IF USING SECANT METHOD, THEN COMPUTE NECESSARY MATRICES APPROXIMATING
C MATRIX OF PARTIALS, BY CONSTRUCTING M INDEPENDENT POINTS LOCALLY
C SURROUNDING X AND USING THESE M+1 POINTS TO DEFINE A HYPERPLANE
C APPROACHING THE ACTUAL TANGENT PLANE AT X --X IS UPDATED AS NEEDED.
C    PERTURB COMPONENTS OF X AND EVALUATE F SUCCESSIVELY. MAINTAIN BEST
C    POINT IN X. UPDATE MINIMAL VALUE,BST, AS REQUIRED. OUTCOME WILL BE
C    M PERTURBED POINTS IN S(NM+N+1),....,S(NM+N+M**2),AND CORRESPONDING
C    FUNCTION VALUES IN S(1),....,S(MN). BEST PT. IS X,BEST FUNCTION OF
C    THIS X IS IN S(MN+1),....,S(MN+N).
  200 K=0
      JD=0
      DO 430 I=1,M
C     SHOULD I-TH COMPONENT OF X BE PERTURBED.
      IF(IV(I))430,210,430
  210 K=K+1
C     PERTURB IT NO LESS THAN (10**(-6))* X(I).
      A1=1.E-6*AB(X(I))
      IF(AB(H(I))-A1) 220,220,230
  220 H(I)=SIGN(A1,H(I))
C     SAVE X(I) TEMPORARILY.
  230 P=X(I)
C     PERTURB X(I) BY H(I).
```

```
  240 X(I)=H(I)+P
C     COMPUTE F FOR NEW POINT.
      CALL FCALC(X,F,S(1),IBLEW)
C     TEST IF POINT IS GOOD. IF NOT, REVERSE H(I),HALVE, AND TRY AGAIN.
      IF(IBLEW)280,250,280
C     COMPUTE NORM SQUARED FOR NEW POINT X.
  250 BF=0.
      DO 260 J=1,N
  260 BF=BF+F(J)**2
C     IF F**2 TOO LARGE, HALVE STEP AND REPEAT.
      IF(BF-1.E36)290,280,280
  280 H(I)=-H(I)/2.0
      GO TO 240
C     UPDATE STEP SIZE AND DIRECTION DEPENDING ON SIGNED CHANGE IN F**2.
C       UP STEP SIZE BY 5 IF RELATIVE CHANGE IN F**2 IS VERY SMALL.
C       DIRECTION OF THIS CHANGE IS TOWARD SMALLEST VALUE OF BF,BST.
 290  IF(AB((BF-BST)/BST)-1.E-7) 300,300,310
 300  H(I)=SIGN(5.0*H(I),(BST-BF)*H(I))
      GO TO 360
C       IS NEW F BETTER.
  310 IF(BF-BST)320,320,340
C       NEW F IS BETTER. IF RELATIVE CHANGE IN F**2 IS LESS THAN .5 '
C       THEN UP STEP SIZE BY 5.
  320 IF((BST-BF)/BST-.5)330,330,360
  330 H(I)=5.0*H(I)
      GO TO 360
C       NEWFNOT BETTER, REVERSE STEP DIRECTION.
  340 H(I)=-H(I)
C       IF RELATIVE CHANGE IN F**2 IS GREATER THAN .5 THEN REDUCE STEP
C       SIZE BY 1/5.
      IF((BF-BST)/BST-.5)360,360,350
  350 H(I)=H(I)/5.0
C     DEFINE START INDEX IN S ARRAY FOR STORAGE OF SUCCESSIVE COMPONENTS
C     OF X WHICH ARE CONSIDERED IN ITERATIVE PROCEDURE.
  360 JZ=JPX+(K-1)*IP
C     SET APPROPRIATE ENTRIES IN S VECTOR TO ABOVE X COMPONENTS.
      K1=0
      DO 380 J=1,M
      IF(IV(J))380,370,380
  370 K1=K1+1
      JJ=JZ+K1
      S(JJ)=X(J)
  380 CONTINUE
C     IS NEW F BETTER. THIS QUESTION IS ASKED TO DETERMINE WHETHER
C     LAST BEST POINT SHOULD BE UPDATED.
      IF(BF-BST)390,390,410
C     NEW F IS BETTER.
  390 JZ=JZ+K
C     PUT PREVIOUS X(I) IN S ARRAY.
      S(JZ)=P
      DO 400 J=1,N
      JD=JD+1
      JK=JPFF+J
C     STORE PREVIOUS FUNCTION VALUE IN APPROPRIATE
```

```
C       SECTION OF S(1),....,S(MN).
        S(JD)=-S(JK)
C       STORE NEW FUNCTION VALUE IN S(MN+1),....,S(MN+N).
  400   S(JK)=-F(J)
        BST=BF
        IF(BST.EQ.0.) GO TO 650
        GO TO 430
C       NEWFNOT BETTER. SET APPROPRIATE SECTION OF S(1),....,S(MN) TO
C       NEW FUNCTION VALUE.
  410   DO 420 J=1,N
        JD=JD+1
  420   S(JD)=F(J)
        X(I)=P
C       REPEAT FOR MORE COMPONENTS.
  430   CONTINUE
C     HAVE SET OF PERTURBED POINTS, AND A BEST POINT. COMPUTE P,Q-MATRICES
C     USED IN SECANT METHOD.
  440   K=0
        K1=0
        DO 480 I=1,IP
C       COMPUTE POINT DIFFERENCES FOR Q-MATRIX,
C       S(MN+N+1),....,S(MN+N+M**2).
        DO 460 J=1,M
        IF(IV(J))460,450,460
  450   K=K+1
        IX=JPX+K
        IQ=JQ+K
        S(IQ)=S(IX)-X(J)
  460   CONTINUE
C       COMPUTE FUNCTION DIFFERENCES IN P-MATRIX,S(1),....,S(MN)
        DO 470 J=1,N
        IPF=JPFF+J
        K1=K1+1
  470   S(K1)=S(K1)+S(IPF)
  480   CONTINUE
        JON=0
        IRX=0
        IRV=JV
C DETERMINE DX= ITERATIVE CHANGE IN X, FOR SECANT METHOD.
C     SOLVE PY = -F  FOR Y.
  490 CALL LEGS(S(1),S(JY+1),BNDS,N,IP,N,IP,
     1S(JATA+1),S(JAR+1),T,SUS,SUP,IBNDS,0,0,0,0,0)
C     COMPUTE DX = QY. STORE IN S(NM+N+M**2+M) SUBVECTOR.
        DO 500 I=1,IP
        IZ=JAZ+I
        S(IZ)=0.
        DO 500 J=1,IP
        IQ=JQ+(J-1)*IP+I
        IY=JY+J
  500 S(IZ)=S(IZ)+S(IQ)*S(IY)
C UPDATE MATRIX ITERATION COUNT FOR THESE P,Q.
        JON=JON+1
        GO TO 520
C DETERMINE DX DIRECTLY,IF USING TANGENT METHOD, FROM  A(DX) = -F ,
C WHERE A IS MATRIX OF PARTIALS STORED IN S(1),....,S(MN) BY SUBROUTINE
C FCALC. STORE DX IN S(NM+N+M**2+M) SUBVECTOR.
  510 CALL LEGS(S(1),S(JAZ+1),BNDS,N,IP,N,IP,
     1S(JATA+1),S(JAR+1),T,SUS,SUP,IBNDS,0,0,0,0,0)
  520 I=0
      DO 540 K=1,M
      IF(IV(K))540,530,540
  530 I=I+1
      IW=JW+I
      IZ=JAZ+I
C SAVE PREVIOUS X,BEFORE THIS LAST MATRIX ITERATION,IN S(NM+N+M**2+2M) S
C SUB-VECTOR.
      S(IW)=X(K)
C COMPUTE NEW X FROM DX AND OLD X.
      X(K)=S(IW)+S(IZ)
  540 CONTINUE
C UP ITERATION COUNT BY 1.
      NIT=NIT+1
      IF(IDER)550,570,550
C SAVE LAST POINT PARTIALS,IF USING TANGENT METHOD. NOTE--A SIMILAR THIN
C WAS DONE FOR Q-MATRIX WHEN USING SECANT METHOD.
  550 DO 560 K=1,JPFF
      IQ=JV+K
  560 S(IQ)=S(K)
C COMPUTE NEW F AND F**2.
  570 CALL FCALC(X,F,S(1),IBLEW)
      BF=0.
      DO 580 I=1,N
  580 BF=BF+F(I)**2
      GO TO 680
C FINAL SEQUENCE OF STATEMENTS ENDING PROGRAM.
C   TEST IF PRESENT NORM SQUARED BETTER THAN LAST.
  590 IF(BF-BST)640,640,600
C   RESET BEST X.
  600 DO 620 I=1,M
      IF(IV(I))620,610,620
  610 JW=JW+1
      X(I)=S(JW)
  620 CONTINUE
C   SET FUNCTION F TO LAST BEST VALUE.
      DO 630 I=1,N
      JPFF=JPFF+1
  630 F(I)=-S(JPFF)
      GO TO 650
C   UPDATE BEST F**2.
  640 BST=BF
C   NULL OUT CONSTRAINT VECTOR.
  650 DO 660 J=1,M
      H(J)=0.0
  660 BNDS(J)=0.0
  670 BSST=BST
      RETURN
C
C
```

```
  680 K=0
C IS POINT GOOD.
       IF(IBLEW)690,700,690
C TEST NO.OF ITERATIONS.
  690 IF(MAXIT-NIT)600,600,840
  700 DO 720 I=1,M
      IF(IV(I))720,710,720
  710 K=K+1
      IZ=JAZ+K
C PERFORM RELATIVE CONVERGENCE TEST.
      IF(AB(S(IZ)).GT.AB(X(I)*EPSL)) GO TO 730
  720 CONTINUE
      GO TO 590
  730 IF(MAXIT-NIT)590,590,740
C IS NEW F BETTER.
  740 IF(BF-BST)750,750,840
C NEW F IS BETTER, DOUBLE VALUES IN CONSTRAINT VECTOR.
  750 DO 760 K=1,IP
  760 BNDS(K)=BNDS(K)*2.0
C UPDATE BEST NORM SQUARED VALUE.
      BST=BF
C IF USING TANGENT METHOD THEN UPDATE BEST F IN
C S ARRAY, I.E., UPDATE CONSTANT MATRIX
      IF(IDER)820,770,820
C UPDATE LAST BEST X--FROM S(NM+N+M**2+2M) TO S(NM+N) SUBVECTOR SECTION.
 770  DO 780 I=1,IP
      IRX=IRX+1
      JJ=JPX+IRX
      IW=JW+I
  780 S(JJ)=S(IW)
      DO 790 I=1,N
      IRV=IRV+1
      IFS=JPFF+I
C SAVE LAST BEST F --FROM S(NM+1) TO OTHER HIGH INDEXED SECTION.
      S(IRV)=-S(IFS)
C UPDATE BEST F -- TO S(NM+1) SUBVECTOR.
  790 S(IFS)=-F(I)
C TEST TO SEE IF P,Q NEED BE UPDATED. IF NO, THEN PERFORM ANOTHER SECANT
C ITERATION.
      IF(JON-IP)490,800,800
C UPDATE S(1),...,S(MN) TO PAST BEST FUNCTION VALUES. TRANSFER TO SECTIO
C WHICH UPDATES P,Q MATRICES.
  800 K=0
      DO 810 I=1,N
      DO 810 J=1,IP
      K=K+1
      IRV=JV+K
  810 S(K)=S(IRV)
      GO TO 440
  820 DO 830 I=1,N
      J=JPFF+I
  830 S(J)=-F(I)
      GO TO 510
C NEW F NOT BETTER.
```

```
 840  IF(IDER) 845,847,845
 845  IF(IBNDS.EQ.0) GO TO 510
 847  I=0
      DO 860 K=1,M
      IF(IV(K))860,850,860
  850 I=I+1
C REDUCE CONSTRAINT VECTOR BY 1/5.
      BNDS(I)=BNDS(I)/5.0
      IW=JW+I
      X(K)=S(IW)
  860 CONTINUE
C RESET FUNCTION F TO LAST BEST VALUES IN S(NM+1) SUBVECTOR.
      DO 870 K=1,N
      JJ=JPFF+K
  870 F(K)=-S(JJ)
      BF=BST
C TRANSFER TO SECTION FOR ANOTHER ATTEMPT AT COORDINATE STEPPING,IF
C USING SECANT METHOD.
      IF(IDER)880,875,880
 875  IF(EPS.GE.0.) GO TO 200
      WRITE(3,8995) JON,NIT
 8995 FORMAT(45H NORM SQUARED WORSE THAN PREVIOUS VALUE AFTER,
     1 I4,26H CONSTANT SLOPE ITERATIONS,10X,
     2 17HTOTAL ITERATIONS=I3)
      GO TO 200
C RESET COEF. MATRIX TO PREVIOUS VALUE, IF USING TANGENT METHOD.
  880 DO 890 I=1,JPFF
      IQ=JV+I
  890 S(I)=S(IQ)
C TRANSFER TO AREA FOR ANOTHER TANGENT ITERATION, WITH THE REDUCED
C CONSTRAINT VECTOR.
      GO TO 510
C START FINISHING PROCEDURE TO TEST WHETHER INPUT X IS MINIMAL.
  900 JPFF=M*N
      JW=JPFF+N
      JZ=JW+M
      WRITE (3,9000)
 9000 FORMAT (44H0 FINISHING PROCEDURE WITH FOLLOWING NOMINAL)
      WRITE (3,9010)(X(IPR),IPR=1,M)
      WRITE (3,9015)(F(IPR),IPR=1,N)
      IF(BF-BST)920,920,930
  920 WRITE (3,9005)BF
      GO TO 940
  930 WRITE (3,9020)BF
  940 DO 1040 I=1,M
      IF(IV(I))1040,950,1040
  950 IF(H(I))970,960,970
  960 H(I)=EPSL*X(I)
  970 P=X(I)
C   PERTURB I-TH COMPONENT,X(I), +H(I) AND -H(I), FOR EACH I. LOOK FOR
C   SMALLER VALUES IN F**2 AND SAVE ACCORDINGLY.
C   TRANSFER TO PROGRAM RETURN.
       DO 1030 J=1,2
      X(I)=P+H(I)
      CALL FCALC(X,S(JZ+1),S,IBLEW)
      SF=0.
      DO 980 K=1,N
      IZ=JZ+K
  980 SF=SF+S(IZ)**2
      WRITE (3,9025)I,H(I)
      WRITE (3,9010)(X(K),K=1,M)
      WRITE (3,9015)(S(K),K=JAR,JATA)
      JAR=JZ+1
      JATA=JZ+N
      IF(BST-SF)1020,990,990
  990 WRITE (3,9005)SF
      DO 1000 JJ=1,M
      IW=JJ+JW
 1000 S(IW)=X(JJ)
      DO 1010 JJ=1,N
      IZ=JZ+JJ
      IPF=JPFF+JJ
 1010 S(IPF)=-S(IZ)
      BST=SF
      GO TO 1030
 1020 WRITE (3,9020)SF
 1030 H(I)=-H(I)
      X(I)=P
 1040 CONTINUE
      GO TO 590
      END
      SUBROUTINE LEGS(AB,DPAR,BNDS,M,N1,MD,ND,ATA,R,Z,
     XSUS,SUSP,I1,I2,I3,I4,I5,IDLE)
C REMOVING THE C FROM THE NEXT TWO CARDS MAKES LEGS DOUBLE PRECISION
C     DOUBLE PRECISION AB,DPAR,BNDS,ATA,R,Z,SUS,SUSP,D1,D2,D
C    1,E1,E2,H,HAL,RHI,RHV,S1,S,SS,X1,X2,X3,X,Y1,Y2,Y3,Y,X1AL,TP
      DIMENSION AB(2),DPAR(2),BNDS(2),ATA(2),R(2),IDLE(2)

C  COUNT THE NUMBER OF DELETIONS

      NIDLE=0
      J=1
    1 IF(IDLE(J))3,3,2
    2 NIDLE=NIDLE+1
      J=J+2
      IF(J-2*M)1,1,3

C  SET UP COUNTERS

    3 N=N1
      N2=N+1
      IQ=(N*N2)/2
      IP=(N2*(N2+1))/2


C  BRANCH ON I3

C     I3 LESS THAN ZERO -- COMPUTE ATA AND ADD IT TO CONTENTS OF ATA
C     I3 EQUAL TO ZERO  -- COMPUTE ATA AND PUT IN CLEARED ATA
C     I3 GREATER THAN ZERO -- ATA CALCULATED BEFORE ENTRY


      IF(I3)6,4,14
    4 DO 5 I=1,IP
    5 ATA(I)=0.
    6 DO 13 K=1,M
      IF(NIDLE)10,10,7

C  ROW DELETOR SECTION

    7 DO 9 J=1,NIDLE
      IS=2*J
      IF(K-IDLE(IS-1))9,13,8
    8 IF(K-IDLE(IS))13,13,9
    9 CONTINUE

C  COMPUTE ATA

   10 L=1
      LI=K
      DO 12 I=1,N2
      LJ=LI
      DO 11 J=I,N2
      ATA(L)=ATA(L)+AB(LI)*AB(LJ)
      L=L+1
   11 LJ=LJ+MD
   12 LI=LI+MD
   13 CONTINUE

C BRANCH ON I5

C       I5 EQUAL TO ZERO      -- CONTINUE
C       I5 NOT EQUAL TO ZERO -- EXIT

      IF(I5)82,14,82

C   BRANCH ON I2

C       I2 EQUAL TO ZERO      -- SOLVE FOR X
C       I2 NOT EQUAL TO ZERO -- DO NOT SOLVE FOR X

   14 IF(I2)15,17,15
   15 DO 16 I=1,N
   16 DPAR(I)=BNDS(I)
      X=1.
      GO TO 71

C   BRANCH ON I1

C       I1 EQUAL TO ZERO      -- NO BOUNDS IN SOLVING FOR X
C       I1 NOT EQUAL TO ZERO -- BOUNDS USED IN SOLVING FOR X

   17 IF(I1)20,18,20
   18 DO 19 I=1,N
   19 BNDS(I)=-1.

C   INITIALIZING FOR ITERATION

   20 RHV=.0001
      RHI=.2
      E1=.1
      E2=.2
      IMAX=20
      ICT=0
      X=0.
      IF=1

C   FIND UNBOUNDED SOLUTION

      GO TO 47

C   IF CONSTRAINTS SATISFIED,EXIT ITERATION
```

```
   21 IF(Y-E1)71,71,22

C   GET FIRST ESTIMATE FOR ITERATION

   22 H=0.
      KT=1
      KL=N2
      IF=2
      K=KT+KL-1
      HAL=0.0
      DO 28 I=1,N
      IF (BNDS(I)) 27,27,23
  23  X1=DABS(ATA(K))
      X1=BNDS(I)**2*(X1/BNDS(I)-ATA(KT))
      X1AL=ATA(KT)*BNDS(I)**2*RHV
      IF(X1AL-HAL)25,25,24
   24 HAL=X1AL
   25 IF(X1-H)27,27,26
   26 H=X1
   27 KT=KT+KL
      KL=KL-1
   28 K=K+KL
      IF(H)29,29,30
   29 H=HAL

C   SET X1=FIRST ESTIMATE

   30 X1=X
      Y1=Y
      X=X+H
      GO TO 47
   31 IF(Y-E1)33,33,32

C   X IS TOO SMALL INCREASE AND TRY AGAIN

   32 H=10.*H
      GO TO 30

C   SOLUTION IS BETWEEN X1 AND X.
C   IF X IS GOOD ENOUGH, EXIT ITERATION

   33 IF(Y+E2)34,71,71
   34 X2=X
      Y2=Y
      IF=3

C   SOLUTION IS BETWEEN X1 AND X2. TRY A POINT IN BETWEEN.

   35 X=(1.-RHI)*X1+RHI*X2
      IF=3
      GO TO 47
   36 IF(Y-E1)37,37,39

C   IF X IS GOOD ENOUGH, EXIT ITERATION
```

```
   37 IF(Y+E1)38,71,71

C  SOLUTION IS BETWEEN X1 AND X.

   38 X3=X2
      Y3=Y2
      X2=X
      Y2=Y

C  SOLUTION IS BETWEEN X1 AND X2. X3 BIGGER THAN X2

      GO TO 40

C  SOLUTION IS BETWEEN X AND X2

   39 X3=X1
      Y3=Y1
      X1=X
      Y1=Y

C  SOLUTION IS BETWEEN X1 AND X2. X3 LESS THAN X1

   40 D1=Y2-Y1

C  DO INVERSE QUADRATIC INTERPOLATION TO GET NEXT POINT

      D2=Y3-Y2
      D=Y3-Y1
      X=0.
      TP=D1*D
      IF(TP)401,402,401
  401 X=X+(X1*Y2*Y3)/TP
  402 TP=D1*D2
      IF(TP)403,404,403
  403 X=X-(X2*Y1*Y3)/TP
  404 TP=D*D2
      IF(TP)405,406,405
  405 X=X+(X3*Y1*Y2)/TP
  406 CONTINUE

C  IF INTERPOLATED POINT IS OUTSIDE RANGE (X1,X2),
C  TRY A NEW POINT BETWEEN X1 AND X2

      IF(X-X2)41,35,35
   41 IF(X-X1)35,35,42

C  INTERPOLATED POINT IS WITHIN RANGE (X1,X2)

   42 IF=4
      GO TO 47

C  IF X IS GOOD ENOUGH, EXIT ITERATION
```

```
   43 IF(Y-E1)44,44,46
   44 IF(Y+E2)45,71,71
   45 X2=X

C  SOLUTION IS BETWEEN X1 AND NEW X2

      Y2=Y
      GO TO 35
   46 X1=X
      Y1=Y

C  SOLUTION IS BETWEEN NEW X1 AND X2

      GO TO 35

C  SOLVING THE LINEAR SYSTEM
C  DECOMPOSE X SUBMATRIX

   47 R(1)=-1.
      R(IP+1)=0.
      L=1
      IF(BNDS(1))49,50,48
   48 R(IP+1)=X/BNDS(1)**2

C  COMPUTE FIRST ELEMENT OF D.

   49 R(IP+1)=ATA(1)+R(IP+1)
   50 DO 62 K=1,N

C  DECOMPOSE (K+1)X(K+1) SUBMATRIX

      IF(K-N)51,53,53
   51 IF(BNDS(K+1))53,52,53
   52 K1=IP+K+1
      R(K1)=0.
      GO TO 61
   53 KL=1
      DO 55 J=1,K
      KM=K+1
      KT=N
      S=0.
      DO 54 I=1,J
      S=S+R(KL)*ATA(KM)
      KM=KM+KT
      KT=KT-1
   54 KL=KL+1
      K1=L+J
      K2=IP+J
      R(K1)=0.
      IF(R(K2))541,55,541
  541 R(K1)=S/R(K2)
  55  CONTINUE
      KL=0
      DO 57 J=1,K
      KL1=KL+J
      S=0.
      DO 56 I=J,K
      K1=L+I
      S=S+R(KL1)*R(K1)
   56 KL1=KL1+I
      K1=L+J
      KL=KL+J
   57 R(K1)=S
      KL=K+1
      KT=N
      S=0.
      DO 58 J=1,K
      K1=L+J
      S=S+R(K1)*ATA(KL)
      KL=KL+KT
   58 KT=KT-1
      K1=IP+K+1
      R(K1)=ATA(KL)-S
      IF(K-N)59,61,61
   59 IF(BNDS(K+1))61,61,60
   60 R(K1)=R(K1)+X/(BNDS(K+1))**2
   61 L=L+K+1
      R(L)=-1.
   62 CONTINUE
      GO TO(63,63,63,63,79),IF
   63 DO 64 I=1,N
      K1=IQ+I
   64 DPAR(I)=R(K1)
      ICT=ICT+1
      IF(ICT-IMAX)65,71,71
   65 S=0.
      DO 67 I=1,N
      IF(BNDS(I))67,67,66
   66 S=S+(DPAR(I)/BNDS(I))**2
   67 CONTINUE
   68 Y=S-1.
   69 Z=X
   70 GO TO(21,31,36,43,79),IF

C     FIND LENGTH OF RESIDUAL VECTOR

   71 SS=0.
      S1=0.
      K1=N2
      K2=N
      DO 76 J=1,N
      L=J
      S=0.
      KK=J-1
      IF(KK)74,74,72
   72 DO 73 I=1,KK
      S=S+ATA(L)*DPAR(I)
   73 L=L+N2-I
   74 DO 75 I=J,N
      S=S+ATA(L)*DPAR(I)
   75 L=L+1
      SS=SS+S*DPAR(J)
      S1=S1+ATA(K1)*DPAR(J)
      K1=K1+K2
   76 K2=K2-1
      SUS=ATA(K1)
      SUSP=SS-2.*S1+SUS


C  BRANCH ON I4

C      I4 EQUAL TO ZERO     -- INVERT NORMAL MATRIX
C      I4 NOT EQUAL TO ZERO -- DO NOT INVERT NORMAL MATRIX,EXIT

      IF(I4)82,77,82
   77 IF(X)78,79,78
   78 X=0.
      IF=5
      GO TO 47

C  FIND THE INVERSE OF (A TRANSPOSE)*A

   79 IRSS=0
      DO 81 K=1,N
      IRSS=IRSS+K
      IRS=IRSS-K
      ITS=IRSS
      DO 81 J=K,N
      IRS=IRS+J
      IT=ITS
      IR=IRS
      S=0.
      DO 80 I=J,N
      IN=IP+I
      IF(R(IN))791,792,791
  791 S=S+(R(IT)*R(IR))/R(IN)
  792 IT=IT+1
   80 IR=IR+I
      R(ITS)=S
   81 ITS=ITS+J
   82 RETURN
      END
```

APPENDIX B

PROGRAM USER'S MANUAL

B.1 INPUT

The input medium for the program is punched cards. The data cards are prefaced with a card containing $AR050 (the program number) in columns 2 through 7 and are terminated by a card with $END in columns 2 through 5. All cards are input using standard FORTRAN NAMELIST (a full description can be found in most FORTRAN manuals).

The NAMELIST variables for each debris piece type are:

MINLD MAXLD CØMNLD SIGMA	L/D variation used to describe the Beta distribution -$L/D)_{min}$, $L/D)_{max}$, $L/D)_{most\ likely}$ and $\sigma_{L/D}$ (which may be omitted - see Section 3.3, No. 2)
CDO CDT CDMAXL	Drag coefficients corresponding to the minimum, most likely, and maximum L/D respectively
ALPHA	Standard deviation of the cg-cp separation in feet (if not given it is set = .01 in the program)
FI	Spin moment of inertia (slug-ft^2)
W	Weight (lb)
AREF	Reference area (ft^2)
ENTRY	The nature of the vehicle's entry. Input 1 for a lunar return and 2 for an earth orbital entry.
TDI	Initial bank angle rate, $\dot{\theta}_i$ ($^\circ$/sec), at breakup (if not input it will be set = 0)

H	Altitude at breakup (ft)
VI	Inertial velocity at breakup (fps)
GAM	Inertial flight path angle at breakup (deg)
AZI	Inertial azimuth at breakup (deg)
BLAT	Breakup latitude (deg)
BLØN	Breakup longitude (deg, positive east of Greenwich)
WCDARF	Ballistic coefficient used to determine the reference point (lb/ft^2) - typically this is the ballistic cofficient of the vehicle as a whole
Y	Confidence ($0 \le Y \le 1$)
NUMAX	Maximum number of iterations allowed (typically 1000)
ERR (extraneous result request)	ERR = 4.0 The Beta distribution and all parameters at each iteration will be printed.
	ERR = 0.0 Or ERR not input, printout will not be produced.

B.2 OUTPUT

The program provides the following output for each piece which is run. The equations used by the program to compute these quantities are given in Appendix E.

REFLAT	The latitude of the reference point
REFLON	The longitude of the reference point
DRBAR	The mean value of all the computed DR's
CRBAR	The mean value of all the computed CR's
SIGDR	The standard deviation of the DR's
SIGCR	The standard deviation of the CR's
COVDC	The covariance of DR and CR
RHO	The covariance coefficient
PSKIP	The percent of total number of cycles in which the piece skipped

NITER	The total number of cycles computed (either at convergence or NUMAX)
EX3	The skew of the DR's
EX4	The kurtosis of the DR's
SIGURF, SIGDRF, XR	The uprange standard deviation, the downrange standard deviation, and the point of split (measured from the reference point) respectively for the best fit of a split bivariate normal distribution to the data

B.3 EXAMPLE INPUT-OUTPUT OPERATION

NL

Date ______ COMPUTATION AND DATA REDUCTION CENTER Page ____ of ____

Name ______ Priority ______

Problem No. ______ NAMELIST INPUT FORM Keypunched by ______

No. of Cards ______ 1 Verified by ______

2 TITLE: 80 COMMENTS

```
$AR050
MINLD     = 0.,
MAXLD     = .355,
CØMNLD    = .1,
CD0       = 1.8,
CDT       = 2.12,
CDMAXL    = 2.189,
ALPHA     = .01,
FI        = 12.1
W         = 244.,
AREF      = 27.65,
ENTRY     = 1.,
H         = 247000.,
VI        = 36356.,
GAM       = -4.65,
AZI       = 85.1,
BLAT      = 32.665,
BLØN      = 159.61,
WCDARF    = 46.932,
P         = .9,
NUMAX     = 2000.,
ERR       = 4.,
$END
```

NL

SYSTEMS 2554 REV. 8-65

```
$AR052

MINLD   =  0.0,
COMNLD  =  0.1E+00,
MAXLD   =  0.355E+00,
CD0     =  0.18E+01,
CDT     =  0.212E+01,
CDMAXL  =  0.2189E+01,
H       =  0.247E+06,
VI      =  0.36356E+05,
GAM     = -0.465E+01,
BLAT    =  0.32665E+02,
BLON    =  0.15961E+03,
AZI     =  0.851E+02,
WCDARF  =  0.46932E+02,
AREF    =  0.2765E+01,
ALPHA   =  0.1E-01,
FI      =  0.121E+02,
W       =  0.244E+03,
TDI     =  0.0,
Y       =  0.9E+00,
NUMAX   =  0.2E+04,




SIGMA   =  0.0,   (NOTE:  0.0 INDICATES THAT SIGMA WAS NOT INPUT)
ERR     =  0.4E+01,
ENTRY   =  1,
$END
```

```
        BETA DISTRIBUTION FOR L/D

A =   1.0000 B =   2.5500
  1 P=   0.000000  X=   0.000000
  2 P=    .010000  X=    .031811
  3 P=    .020000  X=    .048511
  4 P=    .030000  X=    .061407
  5 P=    .040000  X=    .072371
  6 P=    .050000  X=    .082147
  7 P=    .060000  X=    .091105
  8 P=    .070000  X=    .099454
  9 P=    .080000  X=    .107331
 10 P=    .090000  X=    .114829
 11 P=    .100000  X=    .122013
 12 P=    .110000  X=    .128938
 13 P=    .120000  X=    .135649
 14 P=    .130000  X=    .142167
 15 P=    .140000  X=    .148518
 16 P=    .150000  X=    .154721
 17 P=    .160000  X=    .160795
 18 P=    .170000  X=    .166754
 19 P=    .180000  X=    .172612
 20 P=    .190000  X=    .178377
 21 P=    .200000  X=    .184062
 22 P=    .210000  X=    .189673
 23 P=    .220000  X=    .195218
 24 P=    .230000  X=    .200704
 25 P=    .240000  X=    .206138
 26 P=    .250000  X=    .211524
 27 P=    .260000  X=    .216867
 28 P=    .270000  X=    .222172
 29 P=    .280000  X=    .227443
 30 P=    .290000  X=    .232684
 31 P=    .300000  X=    .237898
 32 P=    .310000  X=    .243089
 33 P=    .320000  X=    .248260
 34 P=    .330000  X=    .253415
 35 P=    .340000  X=    .258555
 36 P=    .350000  X=    .263684
 37 P=    .360000  X=    .268805
 38 P=    .370000  X=    .273919


 39 P=    .380000  X=    .279030
 40 P=    .390000  X=    .284140
 41 P=    .400000  X=    .289251
 42 P=    .410000  X=    .294365
 43 P=    .420000  X=    .299485
 44 P=    .430000  X=    .304614
 45 P=    .440000  X=    .309752
 46 P=    .450000  X=    .314903
 47 P=    .460000  X=    .320068
 48 P=    .470000  X=    .325250
 49 P=    .480000  X=    .330451
 50 P=    .490000  X=    .335674
 51 P=    .500000  X=    .340920
 52 P=    .510000  X=    .346192
 53 P=    .520000  X=    .351492
 54 P=    .530000  X=    .356824
 55 P=    .540000  X=    .362188
 56 P=    .550000  X=    .367588
 57 P=    .560000  X=    .373027
 58 P=    .570000  X=    .378508
 59 P=    .580000  X=    .384033
 60 P=    .590000  X=    .389605
 61 P=    .600000  X=    .395227
 62 P=    .610000  X=    .400904
 63 P=    .620000  X=    .406638
 64 P=    .630000  X=    .412433
 65 P=    .640000  X=    .418293
 66 P=    .650000  X=    .424222
 67 P=    .660000  X=    .430226
 68 P=    .670000  X=    .436308
 69 P=    .680000  X=    .442473
 70 P=    .690000  X=    .448728
 71 P=    .700000  X=    .455079
 72 P=    .710000  X=    .461531
 73 P=    .720000  X=    .468092
 74 P=    .730000  X=    .474770
 75 P=    .740000  X=    .481572
 76 P=    .750000  X=    .488509
 77 P=    .760000  X=    .495590
 78 P=    .770000  X=    .502828
 79 P=    .780000  X=    .510233
 80 P=   .790000  X=   .517821
 81 P=   .800000  X=   .525608
 82 P=   .810000  X=   .533612
 83 P=   .820000  X=   .541852
 84 P=   .830000  X=   .550354
 85 P=   .840000  X=   .559144
 86 P=   .850000  X=   .568255
 87 P=   .860000  X=   .577723
 88 P=   .870000  X=   .587595
 89 P=   .880000  X=   .597924
 90 P=   .890000  X=   .608777
 91 P=   .900000  X=   .620238
 92 P=   .910000  X=   .632412
 93 P=   .920000  X=   .645438
 94 P=   .930000  X=   .659499
 95 P=   .940000  X=   .674853
 96 P=   .950000  X=   .691877
 97 P=   .960000  X=   .711153
 98 P=   .970000  X=   .733674
 99 P=   .980000  X=   .761382
100 P=   .990000  X=   .799140
101 P=  1.000000  X=  1.000000
```

```
  L/D        W/CDA        THETA          DR         CR
  .21        35.59        49.54      -18.70      -4.51
  .10        41.34        57.37      -15.57       1.45
  .15        38.56       154.86      -50.90      -1.93
  .09        41.66       347.97        2.00       1.18
  .14        39.01       121.82      -45.25        .57
  .15        38.56       336.32       -2.58       3.78
  .22        35.23       298.30      111.28      17.50
  .04        41.97       328.03        -.33        .61
  .16        38.21        69.54      -34.89       3.08
  .05        41.87        26.22       -4.66        .56
  .09        41.66       308.16        6.86        .71
  .17        37.47        94.64      -46.56       6.89
  .08        41.72       198.01      -18.89      -1.25
  .23        34.84         2.88       14.68       2.80
  .17        37.72       359.53        2.05       7.34
  .04        41.93       233.95       -4.66       -.70
  .08        41.74       129.57      -22.20        .66
  .21        35.59        59.48      -23.98        .16
  .11        40.79       293.40       10.42       1.00
  .11        41.01       132.96      -25.17        .52
  .15        38.56       291.72       15.48       -.74
  .25        33.84       116.28        -.96      14.06
  .15        38.44       210.51      -23.91      -5.55
  .15        38.90       319.85       20.64       3.11
  .12        40.23       266.55       11.49      -3.89
  .19        36.53       181.65      -58.43       1.89
  .07        41.81       266.35         .89      -1.34
  .25        33.84       135.58      -38.38      13.43
  .06        41.84       121.40      -16.20        .35
  .09        41.68       144.25      -26.60       -.23
  .10        41.46        27.47       -6.98       1.27
  .09        41.69        96.75      -21.81        .36
  .03        42.01        40.81       -8.49        .85
  .15        38.90       116.51      -44.87       1.09
  .03        42.01       269.03        -.66       -.48
  .11        41.01       309.69       10.43       2.48
  .16        38.32        59.61      -30.32       3.91
  .21        35.59       249.38       27.06       7.72
  .21        35.76        13.56        2.92      -6.36
  .11        41.23       335.85        4.96       4.01
```

.08	41.75	257.41	1.54	-1.42
.15	38.79	18.30	-13.30	2.88
.08	41.74	331.02	3.52	1.10
.07	41.78	148.32	-23.04	-.00
.12	40.34	334.22	5.97	4.42
.23	34.62	220.40	-30.43	-10.59
.16	37.97	140.06	-54.77	.78
.13	39.90	19.77	-13.30	1.46
.04	41.96	7.38	-4.39	.95
.25	34.12	123.01	-23.31	9.46
.07	41.77	210.62	-15.93	-1.08
.14	39.35	61.51	-28.50	1.24
.07	41.79	229.70	-11.77	-1.03
.11	41.12	203.37	-18.11	-3.93
.15	38.44	247.65	-7.83	-4.12
.08	41.75	133.83	-22.23	.55
.25	33.84	188.93	-15.96	-.82
.15	38.90	39.62	-19.33	3.15
.22	35.42	216.19	2.68	-15.36
.09	41.70	127.62	-23.03	-.39
.15	38.90	168.35	-52.18	-2.50
.04	41.96	353.00	-2.26	.90
.25	33.84	147.65	-25.37	8.30
.18	37.35	257.24	-1.12	-5.85
.18	36.95	188.74	-57.60	-2.24
.19	36.53	201.35	-56.36	-1.20
.05	41.90	227.35	-5.72	-.70
.10	41.34	284.56	15.95	-2.11
.15	38.67	191.54	-31.09	-4.85
.12	40.23	123.83	-37.41	-.75
.17	37.85	83.29	-38.94	8.00
.13	40.01	194.40	-27.06	-4.68
.17	37.47	323.21	.90	.37
.25	34.12	76.89	5.31	12.70
.05	41.88	156.22	-15.80	-.15
.17	37.47	161.04	-57.57	-.19
.07	41.79	240.98	-8.53	-1.02
.12	40.46	81.52	-30.53	.19
.10	41.34	110.92	-23.89	.10
.28	32.59	166.15	-140.71	-3.30
.08	41.74	319.67	4.75	.93
.21	35.76	341.70	25.55	-2.49
.25	33.84	48.71	-52.38	11.11
.17	37.85	347.82	14.68	3.87
.07	41.80	79.96	-11.86	.81
.25	33.84	114.12	-8.12	6.61
.06	41.83	201.29	-15.94	-.68
.06	41.83	280.98	2.33	-1.08
.21	35.93	317.68	28.43	-2.94
.25	33.84	9.20	109.62	9.37
.18	37.22	185.74	-57.21	-2.51
.21	35.59	68.96	-29.36	.17
.09	41.69	294.60	6.04	-.65
.09	41.67	163.27	-26.71	-.48
.08	41.71	340.82	2.91	1.21
.09	41.68	6.33	-2.40	1.05
.05	41.87	70.25	-10.55	.41
.10	41.57	42.06	-10.41	1.84
.09	41.69	104.51	-21.98	.15
.15	38.79	253.06	-6.76	-2.15
.28	32.59	55.12	-39.83	5.39
.13	40.01	235.46	-6.08	-4.78
.14	39.13	272.80	9.18	-2.93
.15	38.90	159.81	-47.58	-2.42
.14	39.01	305.08	17.73	1.61
.06	41.85	123.44	-16.00	.25
.09	41.69	139.73	-26.27	-.08
.20	36.98	124.55	-47.24	-3.66
.09	41.69	277.04	8.85	-.88
.07	41.80	53.32	-7.63	.72
.21	35.93	318.67	29.21	-3.21
.20	36.24	220.43	10.14	4.65
.16	38.21	125.86	-52.34	.70
.09	41.66	108.23	-22.36	.06
.22	35.23	84.88	-41.98	.71
.13	39.79	274.88	8.29	-1.29
.23	34.62	302.50	64.19	1.12
.21	35.76	146.83	-51.35	-4.60
.05	41.88	39.27	-6.57	.56
.22	35.04	319.92	74.42	12.75
.12	40.68	173.46	-27.81	-3.15
.07	41.80	34.32	-3.88	.92

.01	42.10	13.46	-5.28	.14
.13	39.68	333.81	5.49	3.51
.25	34.12	341.43	118.39	6.39
.04	41.96	64.60	-13.67	.55
.08	41.72	136.06	-25.21	.13
.16	38.21	353.90	-11.13	3.33
.18	37.35	27.21	20.51	-5.82
.10	41.65	133.82	-27.28	.00
.16	37.97	20.24	16.16	5.10
.05	41.87	240.67	-7.78	-.62
.20	36.39	312.87	33.20	-5.20
.13	39.79	185.07	-28.18	-4.71
.09	41.70	32.44	-8.26	1.22
.13	39.79	173.18	-31.73	-4.34
.23	34.84	286.17	75.46	-2.89
.23	34.62	124.36	-27.78	-2.30
.18	37.08	322.51	.16	.31
.16	38.09	75.57	-32.67	10.04
.09	41.69	126.20	-23.30	-.34
.10	41.64	287.83	13.81	-1.73
.18	37.35	134.91	-55.61	2.83
.03	41.99	134.99	-17.93	-.28
.19	36.67	330.33	9.62	4.06
.17	37.72	300.27	31.37	-4.14
.24	34.38	281.33	89.13	-2.52
.10	41.57	250.67	1.57	-2.82
.09	41.69	322.35	5.26	.98
.04	41.97	284.80	1.58	-.15
.06	41.84	134.40	-17.04	.25
.14	39.01	214.26	-18.10	-5.16
.15	38.79	41.89	-20.66	3.44
.09	41.66	103.89	-22.69	.18
.10	41.65	302.79	6.41	.57
.18	36.95	261.44	6.25	-4.74
.20	36.24	195.10	-54.08	4.45
.21	35.59	231.55	31.55	-5.43
.11	41.23	187.46	-21.68	-.52
.06	41.84	194.80	-11.63	-.64
.23	34.84	86.53	-46.23	.45
.10	41.46	296.81	18.35	-1.11
.01	42.10	139.22	-12.80	.09

.13	39.68	317.89	8.76	2.35
.22	35.42	232.32	30.11	-12.44
.19	36.67	338.47	-2.00	1.22
.07	41.80	297.53	7.22	-.67
.17	37.47	220.63	-26.92	-5.55
.09	41.66	34.61	-7.73	1.93
.14	39.24	91.08	-42.57	1.80
.05	41.89	70.51	-11.06	.31
.04	41.93	95.85	-17.57	.24
.09	41.66	237.20	-1.20	-1.56
.21	35.93	262.11	1.42	-2.19
.10	41.46	56.23	-14.85	1.48
.08	41.74	267.20	4.96	-1.30
.17	37.72	82.47	-36.96	8.36
.28	32.59	40.50	.52	6.31
.13	39.57	285.28	14.63	-1.07
.07	41.81	8.19	1.19	.98
.22	35.23	329.52	37.36	.42
.05	41.92	196.34	-10.28	-.85
.11	40.90	184.53	-23.40	-3.30
.04	41.94	332.60	-.78	.66
.22	35.42	253.63	28.92	10.58
.02	42.05	251.86	-4.89	-.73
.08	41.76	262.19	2.10	-1.40
.16	38.21	316.07	19.11	2.39
.25	34.12	146.89	-27.26	6.54
.24	34.38	163.67	-26.08	-3.87
.19	36.81	114.08	-45.91	4.88
.16	38.32	268.89	5.69	-3.70
.13	40.01	320.10	8.79	2.56
.18	37.08	36.20	-18.68	7.57
.12	40.12	311.18	8.81	2.42
.05	41.87	240.21	-7.89	-.63
.13	39.90	205.98	-17.56	-4.39
.06	41.83	26.46	-3.54	.68
.09	41.67	248.41	1.63	-1.39
.09	41.69	333.28	4.08	1.14

```
REFLAT =         32.96
REFLON =        165.74
DRBAR  =         -8.52
CRBAR  =           .52
SIGDR  =         30.16
SIGCR  =          4.29
COVDC  =         16.93
RHO    =           .13
PSKIP  =          0.00
NITER  =        200.00
EX3    =      21920.73
EX4    =    6489536.58
SIGURF =         18.95
SIGDRF =         39.88
XR     =        -16.87
```

APPENDIX C

AERODYNAMIC PROPERTIES OF THE DEBRIS

Among the necessary input data required by the program for each debris piece type are values of lift/drag ratio (L/D)

- For the minimum theoretical L/D,
- For the maximum theoretical L/D,
- For the most likely L/D,

and values of drag coefficient (C_D)

- At maximum L/D,
- For the tumbling state,
- And the minimum value at the minimum L/D.

The manner in which these quantities are determined is described in this Appendix.

C.1 NATURE OF DEBRIS

A prerequisite to the analysis performed by this program is a complete state vector at breakup for the vehicle of interest and a list of the resulting debris which is predicted to survive reentry. These various pieces of debris will have been classified into simple shape categories with weights and dimensions assigned to each different debris piece type. While the nature of the debris resulting from an actual vehicle breakup will not result in pieces of such well defined nature, this type of classification is necessary in order to facilitate estimating their aerodynamic properties.

C.2 AERODYNAMIC COEFFICIENTS

The aerodynamic coefficients of the debris are determined for the following limiting conditions: (a) trimmed at zero lift condition, $\alpha = 0^\circ$, $\alpha = 90^\circ$, (b) trimmed at maximum lift/drag ratio orientation, and (c) tumbling (which is considered to be the most probable reentry mode). These aerodynamic coefficients are estimated using Newtonian theory. While it is not possible to determine the actual trim position or the motion of the debris with the limited available mass properties and physical data, these uncertainties are accounted for statistically in the program by randomly selecting the state of a piece, given the limiting values.

Drag coefficients for the trimmed zero lift orientations of $\alpha = 0^{\circ}$ and $\alpha = 90^{\circ}$ are computed using Newtonian theory in which the stagnation point pressure coefficient is chosen as $C_{p_{max}} = 1.8$. A maximum lift-drag ratio for each item of debris is determined from a computation of the lift-drag ratio for a range of angle of attack from $\alpha = 0^{\circ}$ to 90°.

The most probable reentry mode is tumbling and the maximum value of the most probable lift-drag ratio for tumbling reentry was estimated to be $L/D = .1$. Tumbling drag coefficients were determined by a summation of the drag coefficients from $\alpha = 0^{\circ}$ to 90°.

A generalized parametric study has lead to a series of curves which makes it possible to determine these aerodynamic coefficients from the physical data included in the debris list. Figure C-1 gives the zero lift drag coefficients for cylindrical shaped pieces of debris as a function of the length/diameter ratio. Similar data for parallelepipeds, including plates, are given in Figure C-2 as a function of the ratio of side area to reference area. An orientation of $\alpha = 0^{\circ}$ places the minimum area face of the parallelepiped perpendicular to the velocity vector. The maximum area face is taken as the reference area. The maximum L/D and corresponding C_D are presented in a similar fashion for cylinders in Figure C-3 and parallelepipeds in Figure C-4. For cylindrical objects, the tumbling drag coefficients are presented in Figure C-5 as functions of the length-diameter ratio for three modes of tumbling. The tumble axis for each tumble mode is assumed to be oriented perpendicularly to the velocity vector. Similar data is given in Figure C-6 for parallelepipeds. The data in Figure C-6 is representative of tumble modes in which only two of the three face sizes are exposed and one of the faces must be the maximum area face. In order to determine the drag coefficient for the tumble mode involving the minimum area face and middle area face, the data of Figure C-6 can still be used by changing the argument of this figure to A_{min}/A_{middle} and multiplying the resulting drag coefficient by the ratio A_{middle}/A_{ref}.

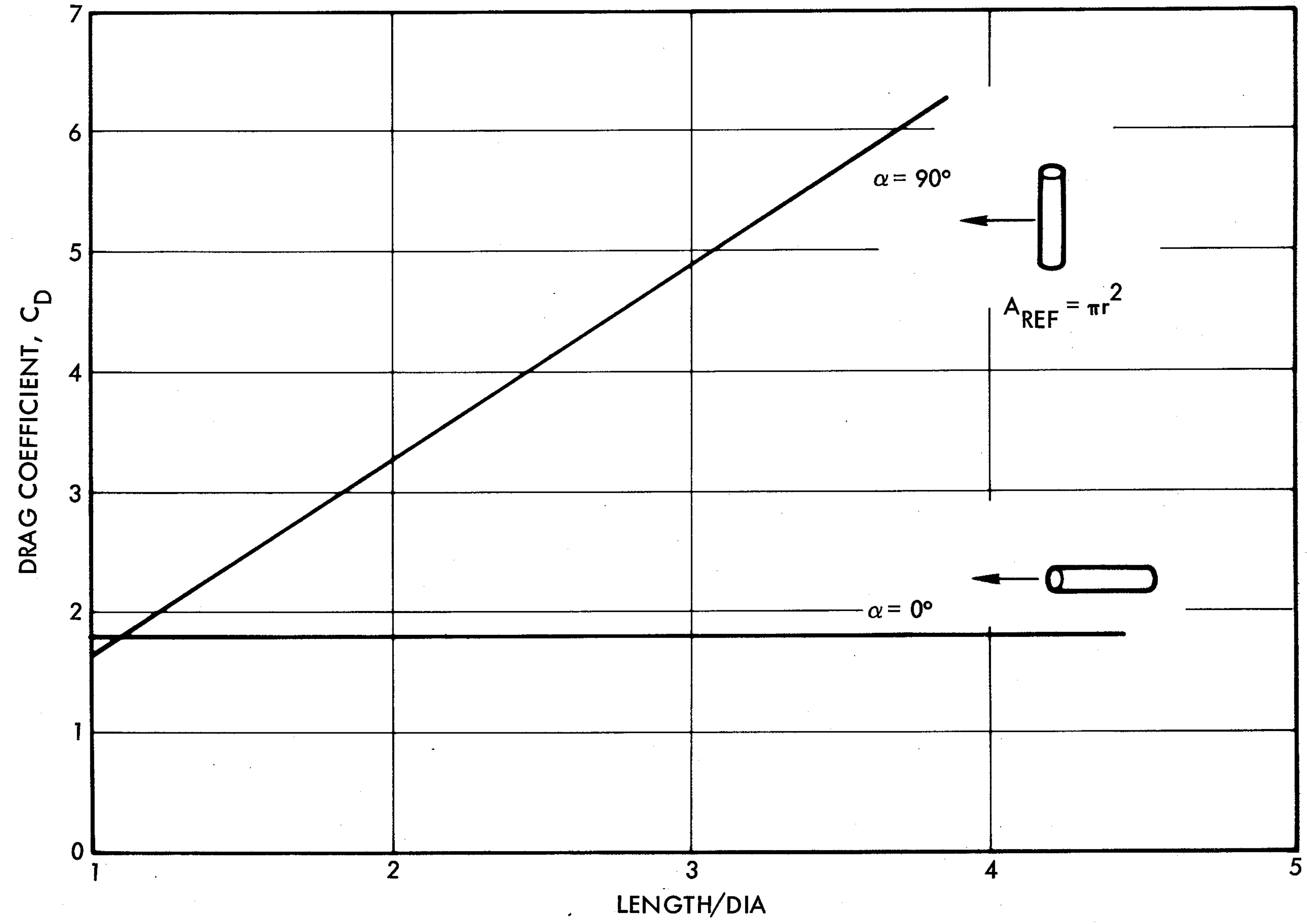


Figure C-1. Drag Coefficients for Cylinders

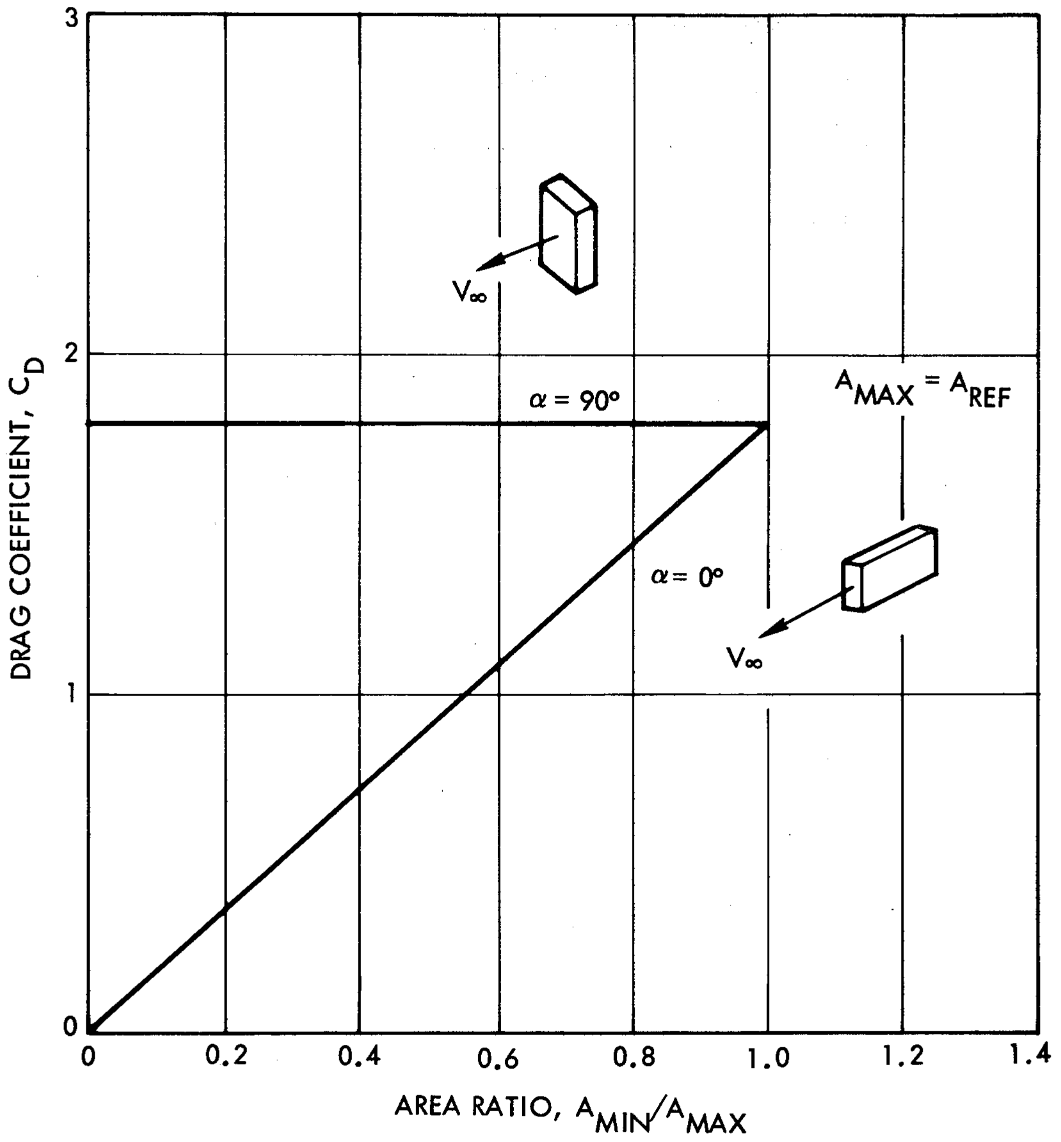


Figure C-2. Drag Coefficient for Parallelepipeds

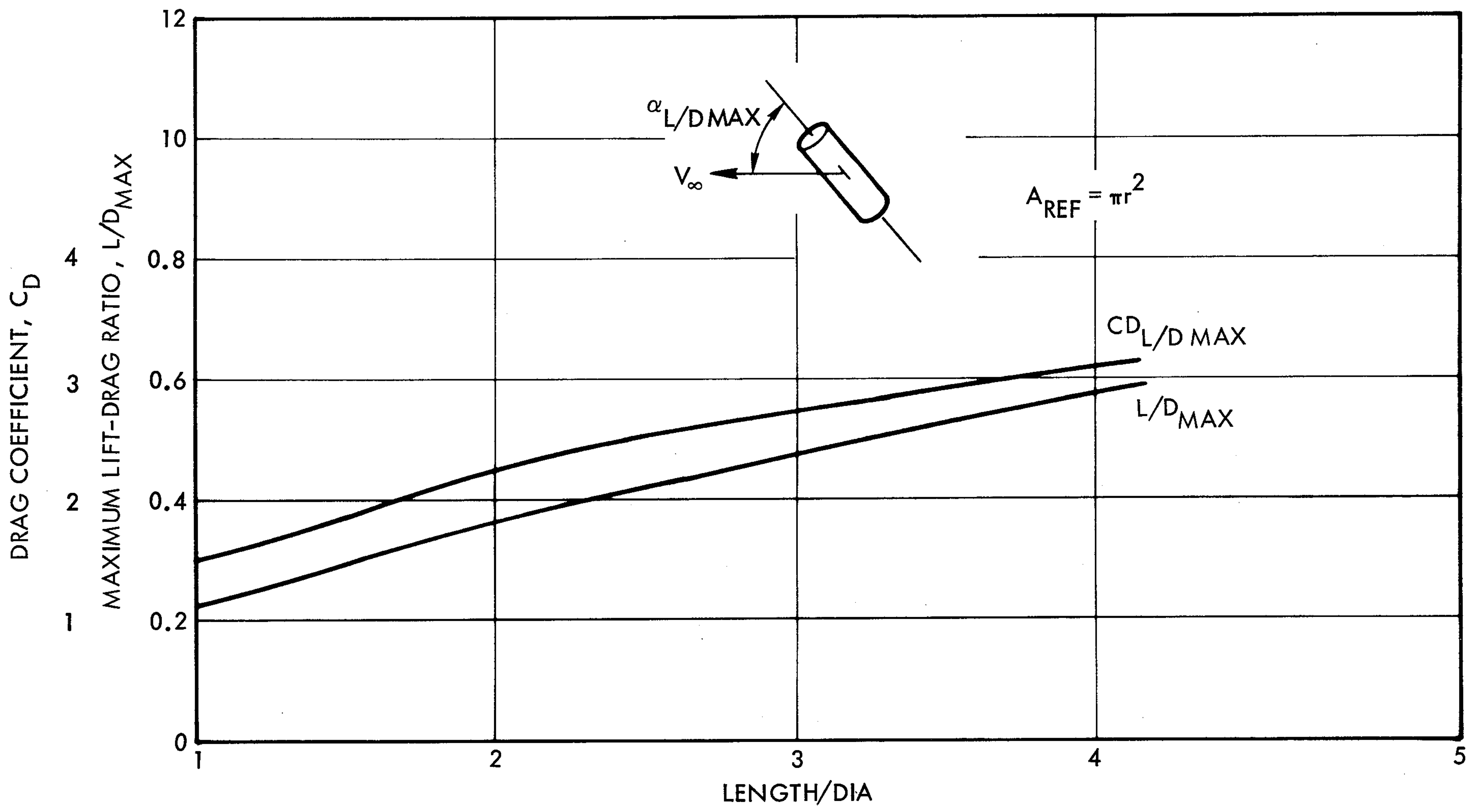


Figure C-3. Maximum Lift-Drag Ratio for Cylinders

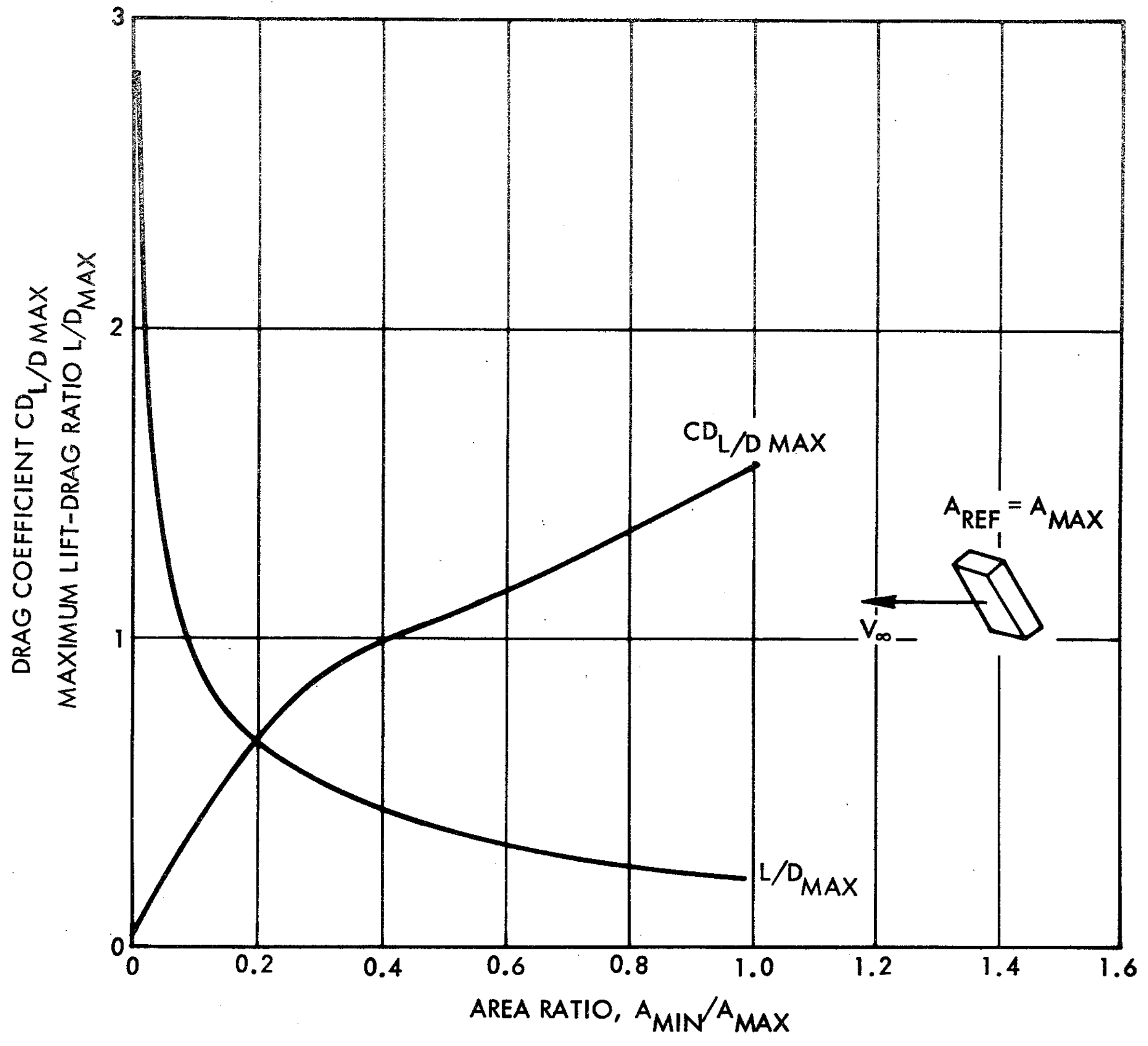


Figure C-4. Maximum Lift-Drag Ratio for Parallepipeds

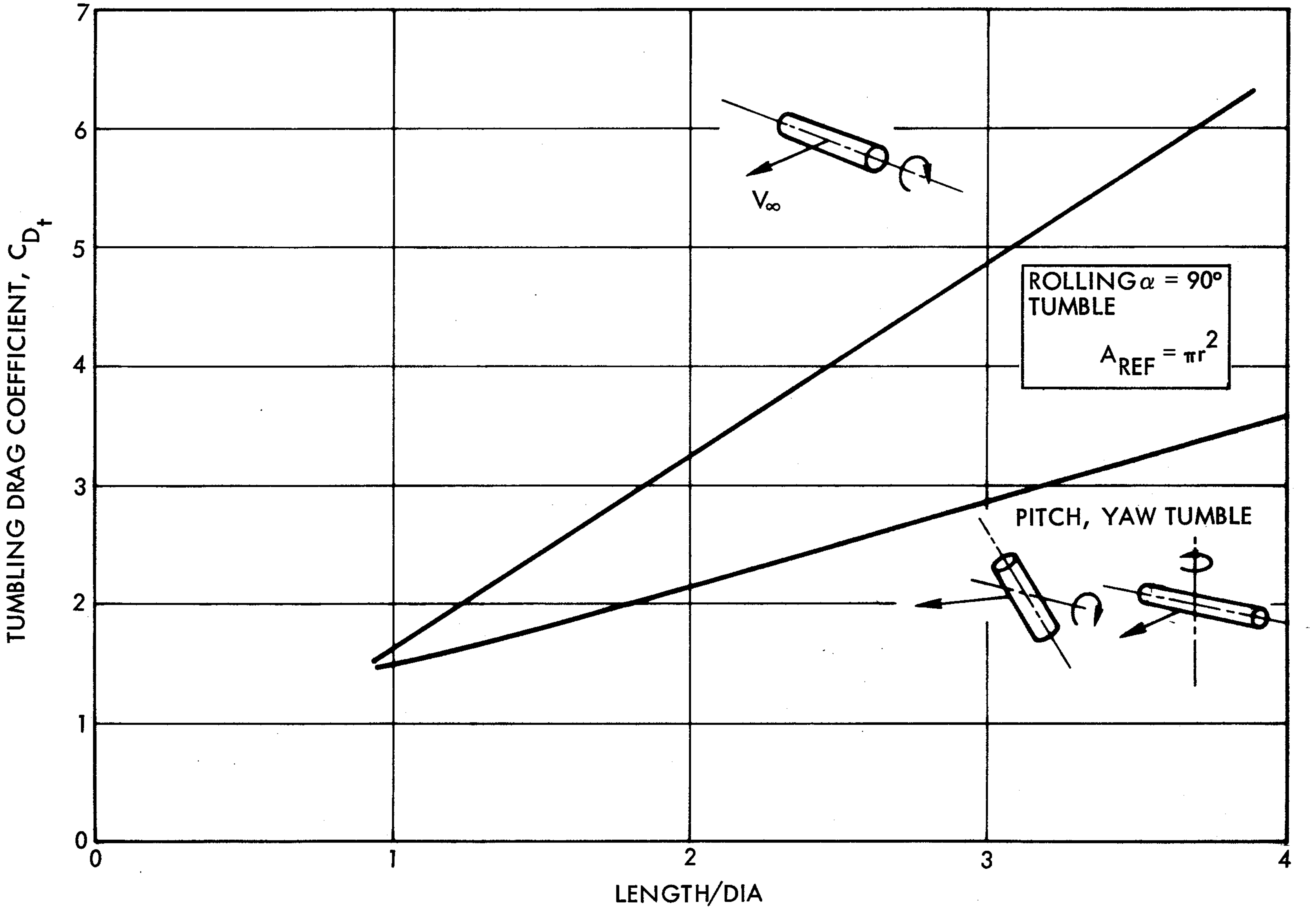


Figure C-5. Drag Coefficient for Tumbling Cylinders

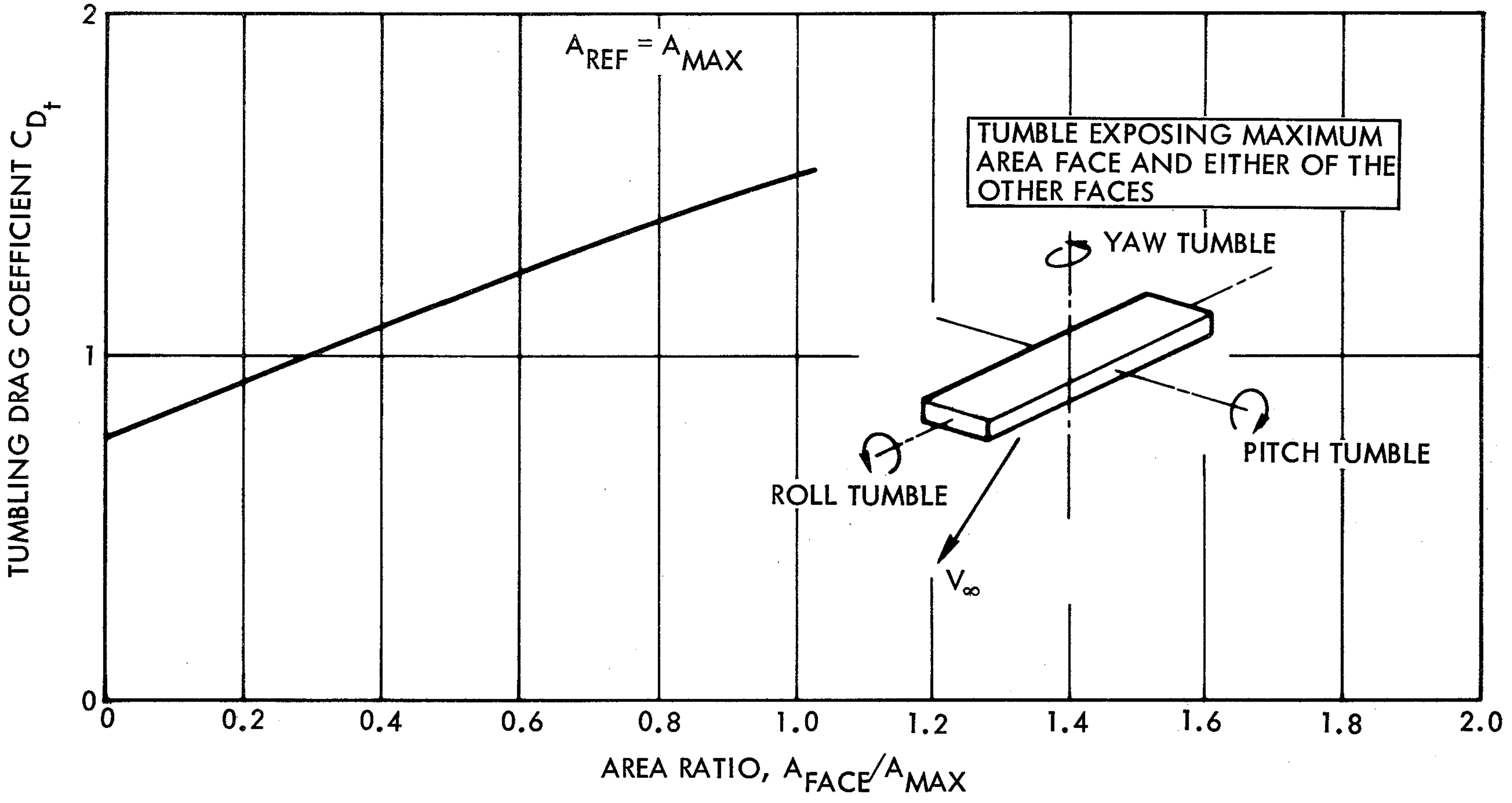


Figure C-6. Drag Coefficient for Tumbling Parallelepipeds

APPENDIX D
REENTRY COMPUTATIONS

D.1 INTRODUCTION

A major effort in the development of this upgraded version of the Apollo Forced Entry Debris Dispersion Program was the formulation of a subroutine to perform a rapid computer calculation of entry trajectories. Emphasis was placed on the speed of computation as the Monte Carlo simulation will call on this subroutine many thousands of times during a mission analysis. The approach which was selected may be called an analytic solution of the entry problem based on a linearization of the equations of motion. The entry of debris is modeled as passing through a rotating exponential atmosphere surrounding a spherical earth and experiencing gravitational, drag, and lift forces.

Such a solution can only be accurate for a limited time duration Δt, and it is necessary to "patch" a series of such solutions together from breakup to impact. Thus, a piece proceeds along the trajectory from a state vector $(\overline{r}_o, \overline{v}_o) = (\overline{r}(t_o), \overline{v}(t_o))$ to $(\overline{r}(t_o + \Delta t), \overline{v}(t_o + \Delta t))$ via one analytic solution $\overline{r}(t)$, $\overline{v}(t)$. The state vector is then "reinitialized" at $t_o + \Delta t$, designated by $(\overline{r}_o, \overline{v}_o)$, and a new analytic solution applied for the next such step of (possibly) different duration Δt. Unlike the case of numerical integration of the equations of motion, the length of the "step-size" interval Δt has not been directly related to the accuracy of the solution as it is propagated. Instead, with a premium on rapidity of calculation rather than on high accuracy, the criterion regulating the size of Δt is based on a rough subjective estimate of the accuracy of the solution rather than on actual predictor-corrector-type calibration of the accuracy of the solution at any given point.

The purpose of this subroutine is to provide for the main program the crossrange and downrange miss distance of a piece of debris with given lift and drag characteristics from a specified reference impact point.

D. 2 CALCULATION OF STATE VECTOR AT BREAKUP

The input to the reentry program is to be a list of values of the following variables (the double-zero subscript refers to values at breakup for which we set t = 0):

(a)
h_{oo} = altitude
v_{oo} = inertial velocity
γ_{oo} = flight path angle
λ_{oo} = latitude of breakup
φ_{oo} = longitude of breakup
ψ_{oo} = azimuth angle

(b) $\overline{r}_{\oplus}$ = nominal impact position vector (in earth-centered inertial coordinates to be described later)

(c)
$\frac{W}{C_D A}$ = ballistic coefficient
$\frac{L}{D}$ = lift/drag ratio
θ_{oo} = angle describing initial lift vector orientation
$\dot{\theta}_{oo}$ = initial bank angle rate of lift vector.
κ = bank angle rate factor.

The variables (a) and (b) are fixed for any one mission analysis. The variables (c) are those which are randomly selected by Monte Carlo methods for each debris piece type. The vector $\overline{r}$ will be determined by a preliminary loop through the subroutine(see section 3.3, No. 19). In this way, any bias in the output due to this particular method of solving the equations of motion will be removed.

The variables (a) yield the state vector ($\overline{r}_{oo}$, $\overline{v}_{oo}$) at t = 0. Expressed in an earth-centered inertial coordinate system with x - axis in the direction of 0^o N, 0^o E at t = 0, y - axis in direction 0^o N, 90^o E at t = 0, and z - axis in the north polar direction.

$$\bar{r}_{oo} = (r_e + h_{oo}) \begin{pmatrix} \cos \lambda_{oo} \cos \varphi_{oo} \\ \cos \lambda_{oo} \sin \varphi_{oo} \\ \sin \lambda_{oo} \end{pmatrix} \quad (r_e = \text{earth radius})$$

and $\bar{v}_{oo} = v_{oo} \bar{u}_{v_{oo}}$, where

$$\bar{u}_{v_{oo}} = \begin{pmatrix} -\sin \varphi_{oo} & \cos \lambda_{oo} \cos \varphi_{oo} & -\sin \lambda_{oo} \cos \varphi_{oo} \\ \cos \varphi_{oo} & \cos \lambda_{oo} \sin \varphi_{oo} & -\sin \lambda_{oo} \sin \varphi_{oo} \\ 0 & \sin \lambda_{oo} & \cos \lambda_{oo} \end{pmatrix} \begin{pmatrix} \cos \gamma_{oo} \sin \psi_{oo} \\ \sin \gamma_{oo} \\ \cos \gamma_{oo} \cos \psi_{oo} \end{pmatrix}$$

It is from this state vector $(\bar{r}_{oo}, \bar{v}_{oo})$ at $t = 0$, then, that an analytic solution is propagated over the first of a series of steps.

D. 3 THE ANALYTIC SOLUTION (t_o to $t_o + \Delta t$)

Outline: Define normalized perturbations, $\bar{p}$ and $\bar{\nu}$, of the state vector near $t = t_o$ by

$$\begin{aligned} \bar{r}(t) &= \bar{r}_o + r_o \bar{p}(t), \\ \bar{v}(t) &= \bar{v}_o + v_o \bar{\nu}(t). \end{aligned} \qquad (1)$$

The equations of motion will next be expanded in terms of the small quantities $\bar{p}$ and $\bar{\nu}$ to terms of the first order.

The acceleration experienced by an entering object is, in this model,

$$\bar{a} = \bar{g} + \bar{d} + \bar{\ell} \qquad (2)$$

where $\bar{g}$ = gravitational acceleration, $\bar{d}$ = acceleration due to drag, and $\bar{\ell}$ = acceleration due to lift. The expressions for $\bar{g}$, $\bar{d}$, and $\bar{\ell}$ are

$$\bar{g} = -\frac{\mu}{r^3} \bar{r} = -g_{SL} \left(\frac{r_e}{r}\right)^2 \left(\frac{\bar{r}}{r}\right) \quad (g_{SL} = \text{gravity at sea-level})$$

$$\bar{d} = -\frac{1}{2} K_D^{-1} \rho \, v_a \bar{v}_a \qquad \left(K_D = \frac{W}{C_D A}\right)$$

$$\bar{\ell} = \frac{1}{2} \eta K_D^{-1} \rho \, v_a^2 \bar{e}_2 \qquad (\eta = L/D, \ \bar{e}_2 = \text{unit vector direction of lift})$$

$$\rho = \rho_{SL} \exp[-\beta(r - r_e)] \qquad (\rho_{SL} = \text{density of air at sea level})$$

Here, $\bar{v}_a = \bar{v} - \bar{\omega}_e \times \bar{r}$ ($\bar{\omega}_e$ = angular velocity vector of earth rotation), is the relative air velocity of the object. Introducing (1) and omitting terms of order higher than p, pν, and ν

$$\bar{g} = \bar{g}_o - g_o\bar{p} - 3(\bar{u}_{ro} \cdot \bar{p})\,\bar{g}_o$$

$$\left(\bar{g}_o = -g_{SL}\left(\frac{r_e}{r_o}\right)^2\left(\frac{\bar{r}_o}{r_o}\right), \ g_o = |\bar{g}_o|, \ \bar{u}_{ro} = \bar{r}_o/r_o\right)$$

$$\bar{d} = \bar{d}_o - \beta(\bar{r}_o \cdot \bar{p})\,\bar{d}_o$$

$$- J_D \left\{ v_{ao} v_o (\bar{\nu} + \bar{\nu}_e \times \bar{p}) + \frac{v_o}{v_{ao}} \left[(\bar{v}_{oo} \cdot \bar{\nu}) \bar{v}_{ao} + (\bar{v}_{ao} \cdot \bar{\nu}_e \times \bar{p}) \bar{v}_{ao} \right] \right\}$$

where

$$J_D = \frac{\rho_{SL} \exp[-\beta(r_o - r_e)]}{2K_D}$$

$$\bar{\nu}_e = -\frac{r_o}{v_o} \bar{\omega}_e$$

$$\bar{d}_o = -J_D \, v_{ao} \bar{v}_{ao}$$

The following relations which are accurate to the first order have also been used:

$$\bar{v}_a = \bar{v} - \bar{\omega}_e \times \bar{r}$$

$$= \overline{v}_o + v_o\overline{\nu} - (\overline{\omega}_e \times \overline{r}_o + r_o\overline{\omega}_e \times \overline{p})$$

$$= \overline{v}_{ao} + v_o(\overline{\nu} + \overline{\nu}_e \times \overline{p})$$

$$v_a^2 = v_{ao}^2 + 2v_o(\overline{v}_{ao} \cdot \overline{\nu} + \overline{v}_{ao} \cdot \overline{\nu}_e \times \overline{p})$$

$$v_a = v_{ao} + \frac{v_o}{v_{ao}}(\overline{v}_{ao} \cdot \overline{\nu} + \overline{v}_{ao} \cdot \overline{\nu}_e \times \overline{p})$$

$$v_a\overline{v}_a = v_{ao}\overline{v}_{ao} + v_{ao}v_o(\overline{\nu} + \overline{\nu}_e \times \overline{p}) + \frac{v_o}{v_{ao}}[(\overline{v}_{ao} \cdot \overline{\nu})\overline{v}_{ao} + (\overline{v}_{ao} \cdot \overline{\nu}_e \times \overline{p})\overline{v}_{ao}]$$

To treat lift, it is necessary to introduce the vectors

$$\overline{u}_1 = \overline{v}_{ao}/v_{ao},$$

$$\overline{u}_{100} = \overline{u}_1 \text{ at } t = 0,$$

$$\overline{u}_{roo} = \overline{u}_{ro} \text{ at } t = 0,$$

$$\overline{n} = \overline{u}_{roo} \times \overline{u}_{100},$$

$$\overline{u}_3 = \overline{u}_1 \times \overline{n}/|\overline{u}_1 \times \overline{n}|,$$

$$\overline{u}_2 = \overline{u}_3 \times \overline{u}_1.$$

The vectors $\overline{u}_1$, $\overline{u}_2$, $\overline{u}_3$ are orthogonal unit vectors. The direction of lift, $\overline{e}_2$, is taken to rotate uniformly about the direction $\overline{u}_1$ in the plane of $\overline{u}_2$ and $\overline{u}_3$ with rate $\dot{\theta}$ where θ is measured clockwise from $\overline{u}_3$ as viewed from the direction of $\overline{u}_1$. Hence

$$\overline{e}_2 = \sin(\theta_o + \dot{\theta}(t - t_o))\,\overline{u}_2 + \cos(\theta_o + \dot{\theta}(t - t_o))\,\overline{u}_3.$$

The bank angle rate, $\dot{\theta}$, is computed prior to each step by

$$\dot{\theta} = \int \ddot{\theta} dt$$

which

$$= \kappa |\bar{\ell}| t + \dot{\theta}_{oo}$$

In the analysis that follows, it is necessary to fix $\bar{e}_2$ in time for the duration of the step $t_o \leq t \leq t_o + \Delta t$. Hence, an average lift of some sort must be used to describe the effect of lift acting for duration Δt, during which time the lift vector rotates through an angle $\dot{\theta}\Delta t$. Rather arbitrarily, the direction of this "effective" lift vector was taken to be that of the actual lift vector at $t = t_o + \Delta t/2$. The "effective" magnitude of $\bar{\ell}$ may be taken to be the time average of the component of $\bar{\ell}$ along the mean direction defined above. If $\bar{\ell}$ were constant in magnitude, it is found that the magnitude of the effective lift vector is that of $\bar{\ell}$ multiplied by a factor

$$\frac{\sin (\dot{\theta}\Delta t/2)}{\dot{\theta}\Delta t/2}$$

Accordingly, through the duration of one step, the direction of lift is constant and given by

$$\bar{e}_2 = \sin (\theta_o + \dot{\theta} (t_o + \frac{\Delta t}{2})) \bar{u}_2 + \cos (\theta_o + \dot{\theta} (t_o + \frac{\Delta t}{2})) \bar{u}_3$$

with an effective L/D ratio of

$$\eta' = \eta \frac{\sin (\dot{\theta}\Delta t/2)}{\dot{\theta}\Delta t/2}$$

The vector $\bar{\ell}$ then becomes

$$\bar{\ell} = \eta' J_D \exp [-\beta (r - r_o)] v_a^2 \bar{e}_2$$

$$= \bar{\ell}_o - \beta (\bar{r}_o \cdot \bar{p}) \bar{\ell}_o$$

$$+ \eta' J_D [2v_o (\bar{v}_{ao} \cdot \bar{v} + \bar{v}_{ao} \cdot \bar{v}_e \times \bar{p})] \bar{e}_2$$

where

$$\overline{\ell}_o = \eta' J_D v_{ao}^2 \overline{e}_2.$$

Then, since $\overline{a} = \dot{\overline{v}} = v_o \dot{\overline{\nu}}$,

$$\dot{\overline{\nu}} = \frac{\overline{g}_o + \overline{d}_o + \overline{\ell}_o}{v_o} \qquad (\equiv \overline{A})$$

$$- J_D v_{ao}[\overline{\nu} + (\overline{u}_1 \cdot \overline{\nu}) \overline{u}_1 - 2\eta' (\overline{u}_1 \cdot \overline{\nu}) \overline{e}_2] \qquad (\equiv \overline{\overline{N}} \cdot \overline{\nu})$$

$$- \frac{g_o}{v_o} \overline{p} - 3 (\overline{u}_{ro} \cdot \overline{p}) \frac{\overline{g}_o}{v_o} - \beta (\overline{r}_o \cdot \overline{p}) \frac{\overline{d}_o}{v_o}$$

$$- J_D v_{ao} (\overline{\nu}_e \times \overline{p}) - J_D (\overline{v}_{ao} \cdot \overline{\nu}_e \times \overline{p}) \overline{u}_1 \qquad (\equiv \overline{f} (\overline{p}))$$

$$- \beta (\overline{r}_o \cdot \overline{p}) \frac{\overline{\ell}_o}{v_o} + 2 \eta' J_D (\overline{v}_{ao} \cdot \overline{\nu}_e \times \overline{p}) \overline{e}_2$$

or

$$\dot{\overline{\nu}} + \overline{\overline{N}} \cdot \overline{\nu} = \overline{A} + \overline{f} (\overline{p}) \qquad (3)$$

where $\overline{\overline{N}}$ is a linear operator, and $\overline{f} (\overline{p})$ is a linear vector function of $\overline{p}$. The operator $\overline{\overline{N}}$ is found to have eigenvectors $\overline{e}_1$, $\overline{e}_2$, $\overline{e}_3$ (unnormalized) where

$$\overline{e}_1 = \overline{u}_1 - 2\eta' \overline{e}_2,$$
$$\overline{e}_3 = \overline{u}_1 \times \overline{e}_2$$

and $\overline{e}_2$ is as above. The associated eigenvalues are

$$N_1 = 2J_D v_{ao}, \quad N_2 = N_3 = J_D v_{ao}.$$

Hence the general solution of the homogeneous equation

$$\dot{\overline{\nu}} + \overline{\overline{N}} \cdot \overline{\nu} = 0$$

is

$$\overline{\nu} = \sum_{1}^{3} K_i \exp\left[-N_i (t - t_o)\right] \overline{e}_i \tag{4}$$

where the K_i are arbitrary constants.

In what follows it will be necessary to transform the components of vectors such as $\overline{\nu}$ with components ν_1', ν_2', ν_3' in the (non-orthogonal) coordinate system $\overline{e}_1$, $\overline{e}_2$, $\overline{e}_3$ into components ν_1, ν_2, ν_3 in the earth-centered inertial system mentioned earlier. It is found that a vector $\overline{b} = b_1' \overline{e}_1 + b_2' \overline{e}_2 + b_3' \overline{e}_3$ has components b_1, b_2, b_3 in the earth-centered system given by

$$\begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} u_{1x} & e_{2x} & e_{3x} \\ u_{1y} & e_{2y} & e_{3y} \\ u_{1z} & e_{2z} & e_{3z} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ -2\eta' & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} b_1' \\ b_2' \\ b_3' \end{pmatrix} \tag{5}$$

and inversely

$$\begin{pmatrix} b_1' \\ b_2' \\ b_3' \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 2\eta' & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u_{1x} & u_{1y} & u_{1z} \\ e_{2x} & e_{2y} & e_{2z} \\ e_{3x} & e_{3y} & e_{3z} \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \tag{6}$$

Next, an approximate particular solution of the non-homogeneous equation is found

$$\dot{\overline{\nu}} + \overline{\overline{N}} \cdot \overline{\nu} = \overline{A} + \overline{f}\,(\overline{p}). \tag{7}$$

It is necessary to find $\overline{p}$ as an explicit function of time and an approximation of constant acceleration $\overline{a}_o = \overline{g}_o + \overline{d}_o + \overline{\ell}_o$ during one step is used to find such an expression:

$$\overline{p}(t) = \frac{\overline{r} - \overline{r}_o}{r_o} = \frac{1}{r_o}\int_{t_o}^{t} \overline{v}\, dt = \frac{1}{r_o}\int_{t_o}^{t} \left[\overline{v}_o + \overline{a}_o (t - t_o)\right] dt$$

$$= \frac{\overline{v}_o (t - t_o)}{r_o} + \frac{\overline{a}_o (t - t_o)^2}{2r_o}$$

Let

$$\overline{B} = \overline{f}\left(\frac{\overline{v}_o}{r_o}\right), \quad \overline{C} = \overline{f}\left(\frac{\overline{a}_o}{2r_o}\right) ;$$

then, since $\overline{f}\ (\overline{p})$ is linear

$$\overline{f}\ (\overline{p}\ (t)) = \overline{f}\left(\frac{\overline{v}_o}{r_o}\right)(t - t_o) + \overline{f}\left(\frac{\overline{a}_o}{2r_o}\right)\left(t - t_o\right)^2$$

$$= \overline{B}(t - t_o) + \overline{C}\ (t - t_o)^2.$$

In coordinates with respect to $\overline{e}_1$, $\overline{e}_2$, $\overline{e}_3$, equation (7) now becomes

$$\dot{\nu}_i' + N_i\ \nu_i' = A_i' + B_i'\ (t - t_o) + C_i'\ (t - t_o)^2$$

(primes indicate vector components in the system $\overline{e}_1$, $\overline{e}_2$, $\overline{e}_3$). One integral of these equations is

$$\nu_i'\ (t) = \left(\frac{A_i'}{N_i} - \frac{B_i'}{N_i^2} + \frac{2C_i'}{N_i^3}\right) + \left(\frac{B_i'}{N_i} - \frac{2C_i'}{N_i^2}\right)(t - t_o) + \left(\frac{C_i'}{N_i}\right)(t - t_o)^2$$

$$= \alpha_i' + \beta_i'\ (t - t_o) + \gamma_i'\ (t - t_o)^2.$$

Combining with the general solution (4) of the homogeneous equation and recalling that $\overline{\nu}\ (t_o) = 0$, the general solution of equation (7) is

$$\nu_i'(t) = \alpha_i' + \beta_i'(t - t_o) + \gamma_i'(t - t_o)^2 - \alpha_i' \exp\left[-N_i(t - t_o)\right]$$

Thus

$$\nu_i(t) = \alpha_i + \beta_i(t - t_o) + \gamma_i(t - t_o)^2 - \sum_j T_{ij}\alpha_j' \exp\left[-N_j(t - t_o)\right] \quad (8)$$

in earth-centered coordinates, where $\|T_{ij}\|$ is the matrix of the transformation (5). Also, since $\overline{v} = r_o \overline{p} = \overline{v}_o + v_o \overline{\nu}$ and $\overline{p}(t_o) = 0$

$$p_i(t) = \int \left(\frac{v_{oi}}{r_o} + \frac{v_o}{r_o}\nu_i\right) dt$$

$$= \frac{v_o}{r_o}\left\{\left(\alpha_i + \frac{v_{oi}}{v_o}\right)(t - t_o) + \frac{\beta_i}{2}(t - t_o)^2 + \frac{\gamma_i}{3}(t - t_o)^3 + \sum_j \frac{T_{ij}\alpha_j'}{N_j}\left(\exp\left[-N_j(t - t_o)\right] - 1\right)\right\} \quad (9)$$

Equations (8) and (9) yield

$$\overline{r}(t) = \overline{r}_o + r_o\overline{p}(t)$$

$$\overline{v}(t) = \overline{v}_o + v_o\overline{\nu}(t) \quad (10)$$

D.4 STEP-SIZE REGULATION

As stated earlier, the choice of a time interval Δt during which the solution (10) is to be applied is not based on the accuracy of the solution after propagating through time Δt; rather, it is based on an estimate of the rate at which the forces acting on the body are changing.

Trial and error with various schemes for step-size control led to the following criterion for the regulation of the step-size Δt. Let Δt be the step-size used in the previous step leading up to time t_o. Evaluate the total acceleration, $\overline{g} + \overline{d} + \overline{\ell}$, at $t_o + \Delta t$ and at $t_o + 2\Delta t$ with state vectors determined by equation (10). Denote these accelerations by $\bar{A}_1$ and $\bar{A}_2$, respectively. Then the value of the quantity $k = |\bar{A}_1 - \bar{A}_2|/|\bar{A}_1|$ = the fractional change in acceleration from $t_o + \Delta t$ to $t_o + 2\Delta t$ is used to control step-size. Specifically if $k < .1$ then the step-size is doubled: $\Delta t \rightarrow 2\Delta t$; if $k > .25$ the step-size is halved: $\Delta t \rightarrow \Delta t/2$. In either case the program cycles through to find a new solution (10) propagated from t_o and then proceeds through the step-size regulation test again. If $.1 \leq k \leq .25$, the same step-size Δt as was used in the previous step is used to propagate from t_o to $t_o + \Delta t$.

The initial step-size (at $t = 0$) is always placed at $\Delta t = 4$ sec.

The following additional constraints are placed on changes in Δt:

1. $1 \text{ sec} \leq \Delta t \leq 16 \text{ sec}$
2. $\Delta t \leq 1/4 \times$ period of roll $= 90^o/\dot{\theta}$
3. Below 100,000 ft Δt is fixed at the last value it had while the altitude was still above 100,000 ft. (It was observed in most cases that at about 50,000 ft the step-size contracted to 1 sec and remained there; at this low altitude the step-size fix could cause no great error in position of impact since the trajectory will be in nearly vertical descent.)

One last modification of the propagation of the trajectory was made to reduce computing time. If at any time the relative velocity vector $\bar{v}_a$ points to within 1^o of the direction of vertical descent, the point on the surface of the earth directly under the piece of debris at that time is taken to be the point of impact; no error larger than a nautical mile or two would be expected from such a modification. In practice this criterion is that descent is considered within 1^o of vertical if

$$\frac{h(t_o) - h(t_o + \Delta t)}{|\bar{v}_a(t_o + \Delta t)|\ \Delta t} > \cos 1^o\ .$$

D.5 IMPACT AND MISS DISTANCE CALCULATIONS

It has been seen how the entry trajectory may be propagated step-by-step using patched analytic solutions. The point of impact of such a trajectory may be determined as follows. Let the reinitialized values of the state vector pass through a sequence

$$(\bar{r}_o, \bar{v}_o)_O, (\bar{r}_o, \bar{v}_o)_1, (\bar{r}_o, \bar{v}_o)_2, \ldots$$

If, at any point, $r_o \leq r_e$ = radius of the earth, then interpolate the analytic solution of the last step before this point to find the state vector at $r = r_e$. In the workings of the actual computer program this is done by fitting a cubic polynomial to the value of r_o as a function of time and solving for $r_o = r_e$. Alternatively, impact may be simulated as described in the last paragraph of the previous section.

In this way a final impact time t_f and position vector $\bar{r}_f$ are found. Let the nominal impact time be t_o and nominal impact position vector be $\bar{r}_\oplus$. To account for the motion of the earth between time t_o and time t_f, rotate the vector $\bar{r}_f$ through the angle $\delta = -\omega_e (t_f - t_o)$ to obtain a vector $\bar{r}'_f$. It is found that

$$\bar{r}'_f = \begin{pmatrix} r_{fx} \left(1 - 2 \sin \frac{\delta}{2} \left|\sin \frac{\delta}{2}\right|\right) - 2r_{fy} \cos \frac{\delta}{2} \left|\sin \frac{\delta}{2}\right| \\ \\ -2r_{fx} \cos \frac{\delta}{2} \left|\sin \frac{\delta}{2}\right| + r_{fy} \left(1 - 2 \sin \frac{\delta}{2} \left|\sin \frac{\delta}{2}\right|\right) \\ \\ r_{fz} \end{pmatrix}$$

Then define unit vectors

$$\bar{k}_1 = \frac{\bar{r}_\oplus}{r_\oplus}$$

$$\bar{k}_2 = \frac{\bar{r}_{oo} \times \bar{r}_\oplus}{\left|\bar{r}_{oo} \times \bar{r}_\oplus\right|}$$

$$\bar{u}_f = \frac{\bar{r}'_f}{r'_f}$$

The vector $\bar{k}_2$ is normal to the plane containing the center of the earth, the breakup point, and the nominal impact point. Thus $\bar{u}_f \cdot \bar{k}_2 = \cos\left(\frac{\pi}{2} - \theta_1\right)$ where θ_1 = the angle between $\bar{r}_f$ and the plane just mentioned. But the crossrange miss distance is

$$CR = r_e \cdot \theta_1 \text{ (radians)} = r_e \cdot \sin^{-1}(\bar{u}_f \cdot \bar{k}_2)$$

Also, if θ_2 = downrange angle of $\bar{r}_f$ measured from $\bar{r}_o$, and ξ = the angle between $\bar{u}_f$ and $\bar{k}_1$, then

$$\cos\xi = \cos\theta_1 \cdot \cos\theta_2$$

and

$$\theta_2 = \cos^{-1}\left\{\frac{\bar{u}_f \cdot \bar{k}_1}{\cos\theta_1}\right\} \qquad \begin{array}{l} \theta_2 > 0 \text{ if } \bar{u}_f \cdot \bar{k}_1 \times \bar{k}_2 > 0 \\ \theta_2 < 0 \text{ if } \bar{u}_f \cdot \bar{k}_1 \times \bar{k}_2 < 0 \end{array}$$

The downrange miss distance is then

$$DR = r_e \cdot \theta_2 \text{ (radians)} = r_e \cos^{-1}\left\{\frac{\bar{u}_f \cdot \bar{k}_1}{\sqrt{1 - (\bar{u}_f \cdot \bar{k}_2)^2}}\right\}$$

D.6 ACCURACY

The preceding formulation has been incorporated as a subroutine to the revised Apollo Forced Entry Debris Dispersion Program. Output from this subroutine has been checked for accuracy against the TRW N-Stage program for a spectrum of lifting and non-lifting reentries with various ballistic coefficients and bank angle rates. The average absolute error for these cases, based on a comparison of impact locations, was on the order of 12 n mi. This would tend to shift the reference point. However, the impact dispersions are based on relative calculations between two impact points computed by the subroutine, and are therefore not affected as strongly by this bias in the program. Considering the uncertainity in the input parameters and the necessity for rapid computation, this reentry solution provides a sufficient degree of accuracy.

APPENDIX E

OUTPUT COMPUTATIONS

The program either computes and stores a DR and CR miss relative to a reference point or registers the occurrence of a skip for each cycle. After n cycles the program has either converged or reached a maximum number of cycles allowed (see Section 3. 3, No. 25) and the program then computes:

Mean values:

$$\overline{DR} = \frac{1}{N}\sum_{i=1}^{N} DR_i \qquad \text{(E-1)}$$

where $N = n - n_s$ (n_s is the total number of skips recorded)

$$\overline{CR} = \frac{1}{N}\sum_{i=1}^{N} CR_i \qquad \text{(E-2)}$$

Standard deviations:

$$\sigma^2_{DR} = \frac{1}{N-1}\left\{\sum_{i=1}^{N} DR_i^2 - N\overline{DR}^2\right\} \qquad \text{(E-4)}$$

$$\sigma^2_{CR} = \frac{1}{N-1}\left\{\sum_{i=1}^{N} CR_i^2 - N\overline{CR}^2\right\} \qquad \text{(E-4)}$$

then $\sigma = \sqrt{\sigma^2}$

Covariance:

$$\text{Cov(DRCR)} = \frac{1}{N-1}\left\{\sum_{i=1}^{N} DR_i CR_i - N\overline{DR}\,\overline{CR}\right\} \qquad \text{(E-5)}$$

Covariance Coefficient:

$$\rho = \frac{\mathrm{Cov(DRCR)}}{\sigma_{DR}\ \sigma_{CR}} \tag{E-6}$$

And for the DR data the following additional output is computed:

Skew:

$$E(x^3) = \frac{1}{N}\sum_{i=1}^{N} (DR_i - \overline{DR})^3 \tag{E-7}$$

Kurtosis:

$$E(x^4) = \frac{1}{N}\sum_{i=1}^{N} (DR_i - \overline{DR})^4 \tag{E-8}$$

In order to interface with existing hazards programs the output also includes the uprange standard deviation, σ_{UR}, the downrange standard deviation, σ_{DR}, and the point of split, x_r, for the best fit to the data of a split bivariate normal distribution. The following derivation shows how these quantities are determined.

The moments of a split normal distribution are given by:

$$E(x^n) = \frac{1}{\sqrt{2\pi}\,\sigma_{UR}} \int_{-\infty}^{o} x^n e^{-\frac{x^2}{2\sigma_{UR}^2}} dx + \frac{1}{\sqrt{2\pi}\,\sigma_{DR}} \int_{o}^{\infty} x^n e^{-\frac{x^2}{2\sigma_{DR}^2}} dx$$

$$= \frac{\Gamma\left(\frac{n+1}{2}\right)}{2\sqrt{2\pi}} \left[\frac{1}{\sigma_{DR}\left(\frac{1}{\sqrt{2}\,\sigma_{DR}}\right)^{n+1}} + \frac{(-1)^n}{\sigma_{UR}\left(\frac{1}{\sqrt{2}\,\sigma_{UR}}\right)^{n+1}} \right] \tag{E-9}$$

If x_r is the distance from the reference point to the split of the distribution and the mean, second moment, and skew (call these M, D, and S, respectively) of the actual data are known, then

$$M = E\left[(x + x_r)\right]$$

$$= E(x) + x_r \qquad \text{(E-10)}$$

$$D = E\left[(x + x_r)^2\right]$$

$$= E(x^2 + 2xx_r + x_r^2)$$

$$= E(x^2) + 2x_r E(x) + x_r^2 \qquad \text{(E-11)}$$

$$S = E\left[(x + x_r)^3\right]$$

$$= E(x^3 + 3x^2 x_r + 3xx_r^2 + x_r^3)$$

$$= E(x^3) + 3x_r E(x^2) + 3x_r^2 E(x) + x_r^3 \qquad \text{(E-12)}$$

Substituting from Equation (E-9) yields the set

$$M = \frac{1}{\sqrt{2\pi}} (\sigma_{DR} - \sigma_{UR}) + x_r \qquad \text{(C-13)}$$

$$D = \frac{1}{2} (\sigma_{DR}^2 + \sigma_{UR}^2) + \frac{2}{\sqrt{2\pi}} x_r (\sigma_{DR} - \sigma_{UR}) + x_r^2 \qquad \text{(C-14)}$$

$$S = \sqrt{\frac{2}{\pi}} (\sigma_{DR}^3 - \sigma_{UR}^3) + \frac{3}{2} x_r (\sigma_{DR}^2 + \sigma_{UR}^2)$$

$$+ \frac{3}{\sqrt{2\pi}} x_r^2 (\sigma_{DR} - \sigma_{UR}) + x_r^3 \qquad \text{(C-15)}$$

The program first computes M, D, and S, and then solves the above set of non-linear cubic equations to determine the σ_{DR}, σ_{UR}, and x_r which give the best fit for the computed data.